Geometric Methods for Mining Large and Possibly Private Datasets

A Thesis
Presented to
The Academic Faculty

by

Keke Chen

In Partial Fulfillment
of the Requirements for the Degree
Doctor of Philosophy

College of Computing
Georgia Institute of Technology
August 2006

Geometric Methods for Mining Large and Possibly Private Datasets

Approved by:

Professor Ling Liu
College of Computing
Georgia Institute of Technology, Adviser

Professor Shamkant Navathe
College of Computing
Georgia Institute of Technology

Professor Elisa Bertino
Department of Computer Science
Purdue University

Professor Edward Omiecinski
College of Computing
Georgia Institute of Technology

Professor Chin-hui Lee
School of Electrical and Computer Engineering
Georgia Institute of Technology

Date Approved: July 6, 2006

To my parents, and my wife

ACKNOWLEDGEMENTS

First of all, I would like to acknowledge the overwhelming contribution and endless hours invested in me by my advisor, Prof. Ling Liu. Without her, I would not even be close to being done. Apart from her research advisory help, I also thank for all the useful life tips I have harvested over the past years from her.

I wish to thank the faculty members on my committee - Prof. Elisa Bertino, Prof. Chin-hui Lee, Prof. Shamkant Navathe, and Prof. Edward Omiecinski - for their help in reviewing the final thesis and giving valuable comments. Prof. Elisa Bertino, Prof. Edward Omiecinski, Prof. Calton Pu, and Dr. Gordon Sun also helped me a lot in my job search. I would like to thank them all.

I am grateful to many people at Georgia Tech for their guidance and support during my Ph.D. study. My colleagues in the Distributed Data Intensive Systems Lab (DISL) were great friends and often provided valuable advice. I especially want to thank Prof. Calton Pu, who had tried to involve me in proposal writing. The DISL group meetings have been a great source of information, pizza, and fun. It was also fun to share experience with my officemates, Bugra Gedik and Bikash Agarwalla. In addition, I wish to thank the administrative staff, Deborah Mitchell, Jennifer Chisholm, and Susie McClain in Center for Experimental Research in Computer Systems (CERCS) for their assistance through the years.

I appreciate the emotional support and encouragement provided by the friends in Atlanta and Buffalo. Zhaohui, Liyu, Yixiang, and I were the famous ”Gang of Four” in Winspear Avenue, Buffalo during 2000-2001 :-). Jun and Yong had helped me settle down when I came to Atlanta in 2001. We have been good fishing and tennis partners since then. I am happy to have you all as my friends.

For my parents, no words can suffice. I offer you my best work with the hope that it stands up to the high standards you have always expected of me. For my wife, Yun, you love and encouragement have been the source of my power and inspiration. To you, I dedicate this thesis.

TABLE OF CONTENTS

LIST OF TABLES

LIST OF FIGURES

SUMMARY

With the wide deployment of data intensive Internet applications and continued advances in sensing technology and biotechnology, large multidimensional datasets, possibly containing privacy-conscious information have been emerging. Mining such datasets has become increasingly common in business integration, large-scale scientific data analysis, and national security. This thesis research addresses three important problems in mining large and possibly private datasets. The first problem is to prove the hypothesis that we can use interactive visualization techniques to develop an effective and yet flexible framework for clustering very large datasets, especially those datasets having irregularly shaped clusters. The second problem is to prove the hypothesis that there is effective method for determining the critical clustering structure of categorical data, i.e., finding the best K number of clusters in categorical data. The third problem is to prove the hypothesis that we can develop multidimensional data perturbation techniques that provide high privacy guarantee with little sacrifice of the accuracy of some data mining models. The proposed research aims at exploring the geometric properties of the multidimensional datasets utilized in statistical learning and data mining, and providing novel techniques and frameworks for mining very large datasets while protecting the desired data privacy.

Some of the most challenging problems in numerical data clustering include identifying irregularly shaped clusters, incorporating domain knowledge into clustering, and cluster-labeling for large amount of disk data. These problems are aggravated when the dataset is huge and the clustering phase is performed on a subset of sampled data. Existing automatic approaches are not effective in dealing with the first two problems, while existing visualization approach does not address the challenges in clustering large datasets. The first main contribution of this research is the development of iVIBRATE interactive visualization-based approach for clustering very large datasets. With the iVIBRATE approach, we address these problems with the visualization-based three-phase framework: “Sampling - Visual Cluster Rendering - Visualization-based Disk Labeling”. The distinct characteristics of the iVIBRATE approach are twofold. (1) We design and develop a VISTA visual

cluster rendering subsystem, which invites human into the large-scale iterative clustering process through interactive visualization. VISTA can effectively resolve most of the visual cluster overlapping with interactive visual cluster rendering. (2) We also develop an Adaptive ClusterMap Labeling subsystem, which offers visualization-guided disk-labeling solution that is effective in dealing with outliers, irregular clusters, and cluster boundary extension for large datasets.

There are many categorical data clustering algorithms having been proposed. However, the important problem of identifying the best K number of clusters is not well addressed yet. The second main contribution is the development of "Best K Plot"(BKPlot) method for determining the critical clustering structures in multidimensional categorical data. The BKPlot method addresses two challenges in clustering categorical data: How to determine the number of clusters (the best K) and how to identify the existence of significant clustering structures. The method has a few unique contributions. (1) The basic method is based on the entropy difference between optimal clustering results with varying Ks. BKPlot can suggest a few candidates for the best K, which identify different layers of critical clustering structures, respectively. (2) We also developed the sample BKPlot theory for characterizing the critical clustering structures in very large categorical datasets. (3) The basic BKPlot method and the sample BKPlot method are extended to characterize the feature of no-cluster datasets, which is then used to identifying the existence of significant clustering structures for a given dataset.

Data perturbation has become popular as a means of privacy-preserving data mining. Most of data perturbation research has been focused on randomization approach, which tries to individually perturb some columns of multidimensional data in order to achieve privacy while preserving the dimensional statistics of these columns. However, there is little effort being made on developing multidimensional perturbation techniques. The third main contribution of this research is the development of the theory of geometric data perturbation and its application in privacy-preserving data classification involving single party or multiparty collaboration. Concretely, the basic theory of geometric perturbation is developed for privacy preserving data classification. It is known that many popular data classification algorithms describe the classification decision boundary as a hyperplane or hyper curved surface in Euclidean space. We can employ random geometric rotation and translation to well preserve the classification boundary and perturb the multidimensional dataset with

high privacy guarantee at the same time. The key of geometric data perturbation is to find a good randomly generated rotation matrix and an appropriate noise component that provides satisfactory balance between privacy guarantee and data quality. We analyze three kinds of inference attacks to geometric perturbation: naive-inference attack, ICA (Independent Component Analysis)-based attack, and distance-based attack. Then, we develop a randomized optimization algorithm to find a good geometric perturbation that is resilient to the above three kinds of inference attacks.

The basic theory of geometric perturbation discusses the challenges in perturbing single-party data. When geometric perturbation is applied to collaborative multiparty data classification, there is an additional challenge: the unification of geometric perturbations. Different parties may prefer different geometric perturbation with high privacy guarantee for their own part of training data. In order to mine a cohesive classification model, all geometric perturbations need to be unified into one geometric perturbation. We study two approaches under the data-mining-service based framework: 1) the ranking approach to agree on one perturbation, 2) the space adaptation approach to transform perturbations to one randomly generated secret perturbation. For each approach, two protocols are developed to address different tradeoffs between the three factors: the privacy guarantee, the data quality, and the efficiency of unifying perturbations.

CHAPTER I

INTRODUCTION

With continued advances in communication network technology and sensing technology, there is an astounding growth in the amount of data produced and made available through the cyberspace. Large datasets have been collected and analyzed in many application domains, varying from Bioinformatics, information retrieval, Physics, Geology, to marketing and business trend prediction. Many have reached the level of terabytes to petabytes [52], and some are also distributed to multiple owners. Managing and analyzing large and distributed datasets of evolving nature to discover important patterns has become a critical task in both business analysis and scientific computing.

In the past decade, many data mining models and algorithms were developed for tackling large datasets and related problems. However, a gap has been growing between the general data mining techniques and their effective deployment by domain experts for a number of reasons. 1) Without a close association of domain knowledge with the data mining models, users are not able to effectively utilize the discovered patterns, or to tune the setting of different parameters to better incorporate their domain-specific knowledge. 2) Social concerns, such as the concerns about privacy-sensitive datasets, have hindered users from collaboratively mining large collection of private datasets.

1.1 Data Visualization and Geometric Methods

As most data mining techniques are developed to automatically extract complex or subtle structures, understanding the data mining models and results has become a serious problem. First, complex or subtle structures are often simplified or approximated by mathematical models to certain extents. While in practice the structures can be very different from dataset to dataset, and from domain to domain, it is difficult for users to understand and validate the simplified/approximated structure generated by algorithms. Second, the automation of structure discovery process distances the user from data exploration process. In many cases, the effective data explorer also needs to understand the subtleties of the process that leads to the data mining results.

Complementary to automatic approaches, data visualization is often used to improve the understanding of data mining models and processes. Moreover, visualization is the important method to display properties of data that have patterns possibly not obtainable by current computation methods [38, 18]. However, challenges exist for effectively visualizing the resulting models or structures for datasets having dimensions higher than three [95, 72, 59, 98]. Particularly, the effective visualization models have to *preserve* certain properties of the resulting models or structures in multidimensional space to help the user understand and validate the models/structures. Since the loss of information happens when the structures are mapped from high dimensional spaces to 2D visual space, many have argued that the composition of multiple visualizations or interactive and dynamically changing visualizations may help people to better understand the structures in multidimensional space [95, 72].

Observing the strong connection between many data mining models and geometric spaces, e.g., the connection between data clustering models and Euclidean spaces, we aim at developing new geometric methods for visualizing data mining models and results. These geometric methods must share two features. First, they are able to preserve (or partially preserve) the important property of data mining model, e.g., the density property of data clustering; second, they can be used to interactively generate continuously changed visualizations. The development of the iVIBRATE framework for clustering large multidimensional datasets (Chapter 3) has shown that we are able to find such geometric methods for a particular data mining model.

Simple visualization methods, such as 2D plots and curves, if well designed, can also be very effective in identifying special structures. Early statistical methods have provided simple visual support for data explorers, such as the cluster validity curves for identifying the best-fit models [57, 83]. In this thesis, we will show a new curve-based cluster validation method for determining the best K number of clusters in clustering categorical data (Chapter 4).

1.2 Privacy Preserving Data Mining and Geometric Methods

With the advances of networking and information technology, collecting and mining data that possibly contains private information has invoked privacy concerns. The U.S. Senate Bill 188, the "Data-Mining Moratorium Act of 2003" [44] is proposed to forbid data-mining (including research

and development) by the entire U.S. Department of Defense, except for searches of public information or searches based on particular suspicion of an individual. In addition, all U.S. government agencies would be required to report to congress on how their data-mining activities protect individual privacy.

There are also many situations in which data providers need privacy protection to perform collaborative mining. For example, biologists may need to mine a large collection of privacy-sensitive patient datasets to perform symptom-based analysis and prediction of a particular disease. Different companies in one retail business sector may be interested in learning the sales of particular hot items, such as iPod sales. They will not join in collaborative mining if the privacy-sensitive data of individual company is not properly protected.

Data perturbation has been one of the popular approaches for privacy-preserving data mining [4, 41]. It protects the private information by changing the original data values while preserving the information critical to the specific data mining model. Obviously, a random change of original data may result in high privacy preservation; but information critical for data mining might get lost as well. Therefore, finding the appropriate trade-off between privacy protection and information preservation is a challenging research issue.

What kind of information do we really need to preserve in data mining? It has been one of the unclear issues in recent research of privacy-preserving data mining. From our initial study, to be task or model specific, and is often multidimensional information [25]. In statistics, the most important multidimensional information is covariance matrix. While the dataset is embedded into Euclidean space, it could be distance (or inner product). Considering many data mining models are tightly related to Euclidean space, we conjecture that geometric transformations that can preserve the multidimensional geometric information would be an effective approach in data perturbation. Our work on geometric perturbation (Chapter 5 and 6) has shown that with random geometric transformation, many classification models can be well preserved, and we can also provide high privacy guarantee for the geometrically perturbed data at the same time.

1.3 Contributions and Organization of the Thesis

Low-dimensional (2D or 3D) geometric methods have been widely applied to multiple computational areas, such as computer graphics and robot motion planning [47]. My Ph.D. research aims at exploring the geometric properties of the multidimensional (usually $>$ 3D) datasets utilized in statistical learning and data mining, and providing novel techniques and frameworks for clustering and mining very large datasets while protecting the desired data privacy. This thesis presents three main contributions of my Ph.D. research.

Some of the most challenging problems in numerical data clustering include identifying irregularly shaped clusters, incorporating domain knowledge into clustering, and cluster-labeling for large amount of disk data. These problems are aggravated when the dataset is huge and the clustering phase is performed on a subset of sampled data. Existing automatic approaches are not effective in dealing with the first two problems, while existing visualization approach does not address the challenges in clustering large datasets. The first main contribution of this research is the development of iVIBRATE interactive visualization-based approach for clustering very large datasets. With the iVIBRATE approach, we address these problems with the visualization-based three-phase framework: "Sampling - Visual Cluster Rendering - Visualization-based Disk Labeling". The distinct characteristics of the iVIBRATE approach are twofold. (1) We design and develop a VISTA visual cluster rendering subsystem, which invites human into the large-scale iterative clustering process through interactive visualization. VISTA can effectively resolve most of the visual cluster overlapping with interactive visual cluster rendering. (2) We also develop an Adaptive ClusterMap Labeling subsystem, which offers visualization-guided disk-labeling solution that is effective in dealing with outliers, irregular clusters, and cluster boundary extension for large datasets.

There are many categorical data clustering algorithms having been proposed. However, the important problem of identifying the best K number of clusters is not well addressed yet. The second main contribution is the development of "Best K Plot"(BKPlot) method for determining the critical clustering structures in multidimensional categorical data. The BKPlot method addresses two challenges in clustering categorical data: How to determine the number of clusters (the best K) and how to identify the existence of significant clustering structures. The method has a few unique

contributions. (1) The basic method is based on the entropy difference between optimal clustering results with varying Ks. BKPlot can suggest a few candidates for the best K, which identify different layers of critical clustering structures, respectively. (2) We also developed the sample BKPlot theory for characterizing the critical clustering structures in very large categorical datasets. (3) The basic BKPlot method and the sample BKPlot method are extended to characterize the feature of no-cluster datasets, which is then used to identifying the existence of significant clustering structures for a given dataset.

Data perturbation has become popular as a means of privacy-preserving data mining. Most of data perturbation research has been focused on randomization approach, which tries to individually perturb some columns of multidimensional data in order to achieve privacy while preserving the dimensional statistics of these columns. However, there is little effort being made on developing multidimensional perturbation techniques. The third main contribution of this research is the development of the theory of geometric data perturbation and its application in privacy-preserving data classification involving single party or multiparty collaboration. Concretely, the basic theory of geometric perturbation is developed for privacy preserving data classification. It is known that many popular data classification algorithms describe the classification decision boundary as a hyperplane or hyper curved surface in Euclidean space. We can employ random geometric rotation and translation to well preserve the classification boundary and perturb the multidimensional dataset with high privacy guarantee at the same time. The key of geometric data perturbation is to find a good randomly generated rotation matrix and an appropriate noise component that provides satisfactory balance between privacy guarantee and data quality. We analyze three kinds of inference attacks to geometric perturbation: naive-inference attack, ICA (Independent Component Analysis)-based attack, and distance-based attack. Then, we develop a randomized optimization algorithm to find a good geometric perturbation that is resilient to the above three kinds of inference attacks.

The basic theory of geometric perturbation discusses the challenges in perturbing single-party data. When geometric perturbation is applied to collaborative multiparty data classification, there is an additional challenge: the unification of geometric perturbations. Different parties may prefer different geometric perturbation with high privacy guarantee for their own part of training data. In order to mine a cohesive classification model, all geometric perturbations need to be unified into one

geometric perturbation. We study two approaches under the data-mining-service based framework: 1) the ranking approach to agree on one perturbation, 2) the space adaptation approach to transform perturbations to one randomly generated secret perturbation. For each approach, two protocols are developed to address different tradeoffs between the three factors: the privacy guarantee, the data quality, and the efficiency of unifying perturbations.

The basic geometric concepts and their relationship to several data mining models are discussed in Chapter 2. Then, three research projects are introduced in the following four chapters. Concretely, Chapter 3 presents the iVIBRATE interactive visualization-based framework for clustering large datasets; Chapter 4 is about the Best K method for determining the best number of clusters in categorical clustering; Chapter 5 and 6 proposes the geometric data perturbation technique for privacy-preserving data classification, and describes its application in multiparty collaborative data classification, respectively. A summary of the PhD research is given in Chapter 7.

CHAPTER II

GEOMETRIC METHODS AND DATA MINING

This thesis presents several novel applications of geometric methods in data mining. In this chapter, we will briefly introduce the basic geometric concepts and methods used in this thesis. Then, we will discuss the connections between some data mining models, which will be used in the later chapters, and the related geometric concepts.

This chapter includes three sections. Section 2.1 introduces the important concepts in Euclidean geometry, Projective geometry and Curves, which are the basis of the iVIBRATE framework (Chapter 3) and geometric data perturbation for privacy-preserving data classification (Chapter 5). Then, Section 2.2 gives the definition and notation of dataset. Section 2.3 describes the relationship between some data mining models and the discussed geometric concepts. More concepts and definitions that are specific to the particular problems will be given in the corresponding chapters.

2.1 Geometric Methods

Let $\mathbb{R}$ represent the real numbers and $\mathbb{R}^k$ be k dimensional real vector space. A k dimensional real vector $\mathbf{u}$ is represented as $\mathbf{u} = [u_1, u_2, \ldots, u_k]^T$, $u_i \in \mathbb{R}$. $\mathbf{u}^T$ means the transpose of the vector. As a convention, we use lower case to represent a real number and bold lower case to represent a vector.

2.1.1 Euclidean Geometric Methods

Euclidean geometry is the most important tool used in this thesis. In this section we will give the definitions of vector, vector space, inner product, distance, and orthogonality. We will also describe several important transformations. Many of the definitions in this section are from the book [47].

Definition 1 (Euclidean Space). *A k dimensional* Euclidean space *is a real vector space $\mathbb{R}^k$ equipped with a symmetric bilinear form $\varphi : \mathbb{R}^k \times \mathbb{R}^k \rightarrow \mathbb{R}$. Specifically, φ is defined by $\varphi(\mathbf{u}, \mathbf{v}) = \sum_i^k u_i v_i$, which is called* inner product.

We also represent inner product with $\langle \mathbf{u}, \mathbf{v} \rangle$ or $\mathbf{u} \cdot \mathbf{v}$, or $\mathbf{u}^T\mathbf{v}$. With the definition, we derive the concept of distance as $\sqrt{\varphi(\mathbf{u}, \mathbf{u})}$, denoted as $\| \mathbf{u} \|$. With the definition of distance, Euclidean space is also regarded as one of the *metric spaces*. The Cauchy-Schwarz inequality is satisfied for inner product and Euclidean distance:

$$|\mathbf{u} \cdot \mathbf{v}| \leq \| \mathbf{u} \| \cdot \| \mathbf{v} \|$$

where $|x|$ is the absolute value of x. We also have the Minkowski inequality.

$$\| \mathbf{u} + \mathbf{v} \| \leq \| \mathbf{u} \| + \| \mathbf{v} \|$$

The concept of orthogonality is based on the definition of inner product. We can define the orthogonal/orthonormal vectors and orthogonal matrix as follows.

Definition 2 (Orthogonal and Orthonormal). *If two vectors,* $\mathbf{u}, \mathbf{v} \in \mathbb{R}^k$*, and* $\mathbf{u} \cdot \mathbf{v} = 0$*, we say the two vectors are* orthogonal. *If* $\| \mathbf{u} \| = \| \mathbf{v} \| = 1$*, they are also* orthonormal.

Definition 3 (Orthogonal Matrix). *A* k *dimensional orthogonal matrix* R *is a* $k \times k$ *square matrix, where all row vectors are orthonormal and all column vectors are also orthonormal.*

With the definition of orthonormal, we have $RR^T = R^TR = I$. I is the k dimensional identity matrix $-$ a $k \times k$ matrix where the diagonal elements are 1 and all other elements are 0.

Then, we can define several important transformations: linear transformation, affine transformation, and orthogonal transformation.

Definition 4 (Linear Transformation). *Given two vector spaces* E *and* F*, a function* $f : E \to F$ *is a linear transformation, if*

$$f(\mathbf{u} + \mathbf{v}) = f(\mathbf{u}) + f(\mathbf{v}), \textit{for any vectors } \mathbf{u} \textit{ and } \mathbf{v} \textit{ in } E$$

$$f(\alpha\mathbf{u}) = \alpha f(\mathbf{u}), \textit{for any real value } \alpha.$$

Definition 5 (Affine Transformation). *Given two vector spaces* E *and* F*, a linear function* $h : E \to F$*, and any vector* $\mathbf{t} \in F$*, the function* $f : E \to F$*,* $f(\mathbf{u}) = h(\mathbf{u}) + \mathbf{t}$ *is an affine transformation.*

Simply, we can treat an affine transformation as a linear transformation plus a translation. For example, the full mapping model used in VISTA visualization model (Chapter 3) is an affine transformation. A strict definition can be found in [47].

An orthogonal transformation is a special linear transformation.

Definition 6 (Orthogonal Transformation). *Given two Euclidean spaces E and F of the same dimension k, a function $f : E \to F$ is an orthogonal transformation if it is linear and*

$$\| f(\mathbf{u}) \| = \| \mathbf{u} \|$$

We also call the above transformation as the "rotation" transformation. Given an orthogonal matrix R and any vector $\mathbf{t}$, note that $f_1(\mathbf{u}) = R\mathbf{u}$ is an orthogonal transformation, while $f_2(\mathbf{u}) = R\mathbf{u} + \mathbf{t}$ is not (it is an affine transformation). The transformation f_2 will be discussed in Chapter 5 as the main method to perturb datasets in order to preserve better data privacy and data quality.

For orthogonal transformations, we also have the following properties, which will be used in discussing the rotation-invariant classifiers in Chapter 5.

Lemma 1. *Given any two Euclidean spaces E and F of same dimension k, for every function $f : E \to F$, the following properties are equivalent:*

1. *f is a linear map and $\| f(\mathbf{u}) \| = \| \mathbf{u} \|$, for all $\mathbf{u} \in E$.*
2. *$\| f(\mathbf{u}) - f(\mathbf{v}) \| = \| \mathbf{u} - \mathbf{v} \|$, for all $\mathbf{u}, \mathbf{v} \in E$, and $f(0) = 0$*
3. *$f(\mathbf{u}) \cdot f(\mathbf{v}) = \mathbf{u} \cdot \mathbf{v}$, for all $\mathbf{u}, \mathbf{v} \in E$.*

f_1 discussed above will preserve both distance and inner product, while f_2 can preserve only distance.

In addition, we also define the scaling transformation, which will be used in later discussion. Let $s_i \neq 0, s_i \in \mathbb{R}$. A scaling matrix S is defined by

$$S = \begin{bmatrix} s_1 & & & \\ & s_2 & & \\ & & \ddots & \\ & & & s_k \end{bmatrix}$$

Definition 7 (Scaling Transformation). *A scaling transformation is a linear transformation f, $f(\mathbf{u}) = S\mathbf{u}$, and S is a scaling matrix.*

2.1.2 Contracting projection

The complete definition of projection can be found in the book [47]. In this subsection, we primarily discuss the concept of "contracting projection", which is related to our VISTA visualization model. However, the following definition and Figure 1 is from the paper [59].

Definition 8 (Contracting Projection). *Given two Euclidean spaces E and F, a contracting projection is defined by a linear transformation, $f : E \rightarrow F$, so that*

$$\| f(\mathbf{u}) \| \leq \| \mathbf{u} \|$$

Intuitively, after contracting projection, distances between the points are preserved or reduced. Figure 1 shows the difference between contracting and non-contracting projections. Paper [59] also proves that contracting projection strictly preserves the density in multidimensional space.

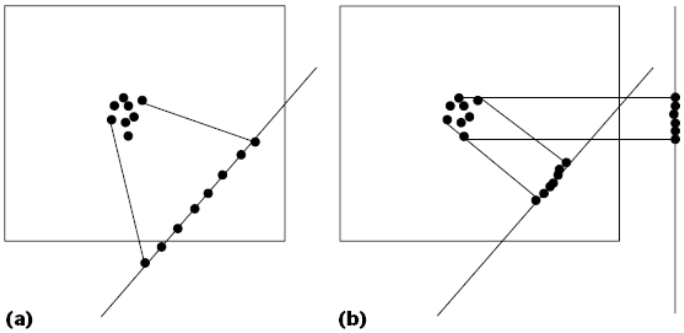

Figure 1: (a) shows a non-contracting projection, while (b) shows contracting projections. Intuitively, contracting projections will preserve the density based on Euclidean distance.

2.1.3 Differential of Curve

We define the two-dimensional (2D) curve in Euclidean space and discuss the meaning of the first order differential and the second order differential of curve, which will be used in developing the Best K method in Chapter 4.

Definition 9 (2D Open Curve). *A two-dimensional open curve is defined as a continuous map: $f : (a, b) \rightarrow \mathbb{R}^2$, where (a, b) is an open interval (allowing $a = -\inf$ and $b = +\inf$). If all the*

derivatives $f^{(k)}$ exist and are continuous for all k, $0 \leq k \leq p$ (when $p = 0$, $f^{(0)} = f$), f is also of class C^p.

We briefly describe the meaning of $f^{(1)}$ and $f^{(2)}$. $f^{(1)}(t)$ determines the *tangent line* (or the changing rate) at the value t, $t \in (a, b)$. $f^{(2)}$ helps to find the "knees" on the curve f, where the value $f(t)$ changes dramatically in the local area of t. Concretely, peaks and valleys of $f^{(2)}$ indicate the corresponding "knees" on f (Figure 2). In Chapter 4, since the entropy difference curve is decreasing, we plot $-f^{(2)}$ (in discrete form, in fact) to make the peaks correspond to the knees.

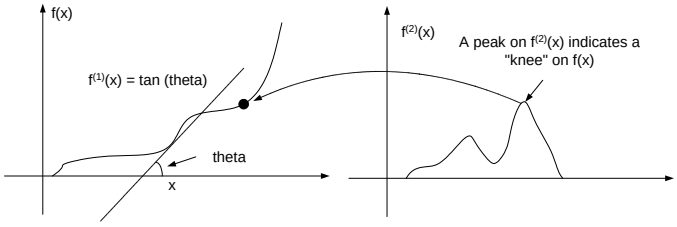

Figure 2: Differentials of curve.

2.2 Definition of Datasets

A dataset in data mining or statistical analysis is usually defined as a set of multidimensional vectors, represented as a d column(dimension) and N row (instances, or records). For the convenience of mathematical manipulation, we use $X_{d\times N}$ to denote the dataset, i.e., $X = [\mathbf{x}_1 \dots \mathbf{x}_N]$, where $\mathbf{x}_i$ is a data tuple, representing a vector in the real vector space $\mathbb{R}^d$. When the dataset is a training dataset in classification modeling, each data tuple $\mathbf{x}_i$ belongs to a predefined class, which is indicated by the class label attribute y_i. The class label can be nominal (or continuous for regression modeling).

For categorical dataset X, we define the values in column j are from the domain A_j. A_j is conceptually different from $A_k (k \neq j)$. There are a finite number of distinct categorical values in $domain(A_j)$ and we denote the number of distinct values as $|A_j|$.

A dataset X can also be understood as a set of d-dimensional points in the d-dimensional Euclidean space. Therefore, all Euclidean geometric concepts we defined earlier can be applied. A dataset can also be embedded into other metric spaces, but it is rare to do so.

If we consider X is a sample dataset from the d-dimensional random vector $\mathbf{X} = [\mathbf{X}_1, \mathbf{X}_2, \dots, \mathbf{X}_d]^T$, most multidimensional statistical methods [76] study the statistical property of such a dataset. Since

the basic statistical methods are based on metric spaces, there is an inherent connection between statistical methods and geometric concepts.

The term “large datasets” is also frequently used in this thesis. A dataset can be large in terms of either large number of records or of high dimensionality, or of both. Particularly, “large” means a scale that current computer hardware/software system cannot comfortably handle. For example, the number of records is so large, that algorithms in non-linear space/time complexity cannot handle them with current hardware configuration; or, the number of dimensions is so large that current visualization system and display devices cannot handle with them. In this sense, a dataset having millions of records, or thousands of dimensions, would be considered as a large dataset. For such large datasets, we have to develop linear or sub-linear complexity algorithms to process them. Statistical methods, such as sampling or summarization, can also be applied to reduce the scale of data for efficient data mining or statistical analysis.

As a convention, we use bold lower cases to represent vectors, lower cases to scalars, bold upper cases to random variables, and upper cases to matrices.

2.3 Data Mining Models and Their Geometric Meaning

Two data mining models are primarily studied in this thesis. The first one is data clustering, including clustering numerical data and clustering categorical data. The other model is data classification, especially the data classification models that look for geometric decision boundary. Concretely, challenges in three specific areas are addressed by this thesis: (1) the challenges in clustering large numerical datasets that have irregularly shaped clusters and domain-specific clustering structures; (2) the challenges in identifying the critical clustering structures for categorical data, specifically, the problem of Best K number of clusters; (3) the challenges in providing better data perturbation techniques that can better preserve both the data privacy and the model accuracy for data classification models.

2.3.1 Density-based Data Clustering

Data clustering is the unsupervised classification of patterns (observations, or feature vectors) into groups (clusters) with certain similarity measure [66]. For numerical data, the most popular similarity measure is Euclidean distance. By applying Euclidean distance, the data records become points, and the groups of points (the clusters) become the dense areas in the Euclidean space. Therefore, density estimation is also an effective and intuitive means doing numerical data clustering. Typical density-based algorithms include DBSCAN [39], OPTICS [8], and DENCLUE [58]. By associated with the concepts of distance and density, data clusters can have certain shapes. In typical statistical methods, such as mixture models [57], the shape of cluster is assumed to be spherical (matched by the assumption of multivariate normal distribution). Therefore, when such an assumption is not held in real datasets, large errors can happen. Clustering datasets having irregularly shaped (non-spherical) clusters is regarded as a difficult problem.

Due to the density property of data clustering, cluster visualization has become the most intuitive method for validating irregularly shaped clusters. Cluster visualization has multiple advantages and can potentially solve some challenges that algorithms and statistical methods cannot handle. However, it is well known that multidimensional data visualization is also a difficult problem.

HDEyE [59] shows that projective mappings between k-dimensional (KD, $k > 3$) space and 2-dimensional (2D) visual space can preserve the density property of the dataset and thus effective in visualizing clusters. Concretely, contracting projection can strictly preserve the density of dataset. Therefore, if a cluster boundary is found to separate two groups of points in 2D cluster visualization generated by contracting projection, the two groups are also separated in the original data space. The only visual bias caused by contracting projection is visual cluster overlapping $-$ two clusters are mapped to the same visual area.

Contracting projection is only a sufficient condition for preserving the density. There are other transformations can preserve density, too. In visualization, a zooming operation can be defined as a scaling transformation. Based on the following observation,

Observation 2. *Zooming a 2D cluster visualization generated by contracting projection will not change the separation of visual clusters.*

we know that the combination of zooming (scaling transformation) and contracting projection can also be used to visualize the density property of dataset. α-mapping developed in Chapter 3 is such a transformation.

α-mapping also has other advantages, such as dimensionally tunable parameters and dynamic properties. Most importantly, with α-mapping and VISTA system, we show that automatically finding visual cluster separators, such as HDEyE [59] and projection pursuit [57], is not necessary and not effective. Instead, we can design an interactive dynamic visualization system that fully utilizes human's unique ability of visual pattern recognition to efficiently find satisfactory cluster visualizations.

2.3.2 Data Classification Models

Data classification is also called *statistical learning from data*. In a typical scenario, we have an outcome measurement, usually quantitative (like a stock price) or categorical (like disease/no disease), that we wish to predict based on a set of features (like diet and clinical measurements) [57]. Typical data used in classification modeling is called training data (see Section 2.2), in which we observe the outcome (the class label) and feature measurements for a set of objects (instances, or data records). Using this data we build a predictive model, also called a classification model or a classifier.

The existing basic data classification models [57] can be approximately categorized by decision tree methods, Bayesian methods, kernel methods, support vector machines (SVMs), linear classifiers, and neural networks. Each category may include multiple concrete models. Among these categories, many models search for geometric decision boundary, such as kernel methods, SVMs, and linear classifiers. We name them "Geometric Classifiers". In Chapter 5 and 6, we will discuss the geometric data perturbation methods specifically designed for privacy preserving data classification with such geometric classifiers. In this section, we roughly describe the geometric meanings behind these models, and why geometric perturbation works for them.

Often, we can embed the training dataset into the Euclidean space and try to find if the points in the same class are also geometrically close to each other, and the points in different classes should be distant from each other. Similar to distance-based clustering, distance works as the key metric

here. We take K Nearest Neighbor (KNN) classifier [57] as example. Given a point $\mathbf{u}$, we predict its class label by looking at the K nearest points in the training dataset. $\mathbf{u}$ is labeled by the class that contains the majority of the K points. This classification model is solely determined by Euclidean distance measure. Similar classifiers include kernel methods, where the training points in the local area of the given point are critical to make decision.

There are some classifiers, such as Perceptron [57], also looking for hyperplanes in multidimensional Euclidean space to separate the classes in training data. These classification models are typically called linear classifiers. The key component of linear classifier is the training process that finds the hyperplanes to separate the classes. For example, the algorithm used in training Perceptron uses *stochastic gradient descent* to minimize the error rate of classification. Some training algorithms look for the optimal separating hyperplane that separates the classes and maximizes the distance between the hyperplane and the separated classes [33], which results in the *support vector* based classifier.

Extending the method using in training support vector classifier with the concept of kernel function, we get the popular *support vector machine* classifiers [33]. Kernel functions (see Chapter 5, Section 5.4) are typically defined using inner product or Euclidean distance.

We have briefly reviewed some geometric classifiers and their connection to geometric spaces. As we have shown in the first section in this chapter, geometric properties, such as distances and inner products, can be preserved under certain geometric transformations. In Section 5.4, we show that these geometric classifiers can also be *preserved* under certain transformations. The problem of geometric perturbation is thus transformed to finding a *good geometric transformation*, with which the perturbed data is more resilient to attacks.

A geometric perturbation can be treated as a one-way function. A one-way function is a function that is easy to evaluate but computationally difficult (NP hard) to invert [50]. One-way function is the basis of many modern encryption algorithms, such as RSA [71]. Typical one-way function includes integer factorization, decoding of random linear codes, and subset sum [50]. In Chapter 5, we will study whether geometric perturbations are strong one-way functions, and, if they can be possibly weakened by knowledge enhanced attacks, how to improve the security of geometric perturbation.

CHAPTER III

IVIBRATE: INTERACTIVE VISUALIZATION BASED FRAMEWORK FOR CLUSTERING LARGE DATASETS

This chapter presents the first piece of the thesis research. This research proves the hypothesis that we can use interactive visualization techniques to develop an effective and yet flexible framework (iVIBRATE) for clustering very large datasets, especially those datasets having irregularly shaped clusters. This research explores the geometric density property of clusters in Euclidean space, and develops novel interactive visualization techniques (VISTA) to dynamically visualize the clusters in multidimensional datasets. VISTA interactive visualization system is also extended to process large datasets with visualization-based labeling system (ClusterMap). The two subsystems make up the iVIBRATE framework. We will show that the iVIBRATE framework is effective and efficient in reducing the error rates of handling large datasets.

This chapter is organized as follows. First, we review the general problems with clustering large datasets in Section 3.1 and sketch the scope and contributions of this chapter in Section 3.2. Section 3.3 reviews some related work. Section 3.4 presents the design principles and components of iVIBRATE framework. Section 3.5 and Section 3.6 briefly introduce the two main components of the framework: the VISTA visual cluster rendering system and the ClusterMap labeling algorithms. Section 3.7 describes the problems and the solutions in the integration of the three phases. Section 3.8 reports some experimental results, demonstrating that the iVIBRATE approach can not only effectively discover and validate irregular clusters, but also effectively extend the intermediate clustering result to the entire large dataset. We also present an detailed example, showing how to explore a very large real dataset: census dataset with the iVIBRATE framework, in section 3.9.

3.1 General Problems with Clustering Large Datasets

Several clustering algorithms have aimed at processing the *entire* dataset in linear or near linear time, such as WaveCluster [94], DBSCAN [39], and DENCLUE [58]. However, there are some

drawbacks with these approaches.

(1)Time Complexity of Iterative Cluster Analysis. Typically, cluster analysis continues after the clustering algorithm finishes in a run, unless the user has evaluated, understood and accepted the clustering patterns or results. Therefore, the user needs to be really involved in the iterative process of "clustering and analysis/evaluation". In this process, multiple clustering algorithms, or multiple runs of the same algorithm with different parameter settings can be tested and evaluated. Even for a clustering algorithm with linear computational complexity, running such an algorithm on a very large dataset for multiple times could become intolerable. Moreover, most cluster validation methods have non-linear time complexity [56, 66, 37]. When performed on the entire large dataset, the validation of clusters hinders the performance improvement for the entire iterative process.

(2)Cluster Analysis on Representative Dataset vs. on Entire Dataset. Bearing the above problems in mind, a number of approaches were proposed to perform clustering algorithms on smaller sample datasets or data summaries instead of the entire large dataset. For example, CURE [53] applies random sampling to get the sample data and then runs a hierarchical clustering algorithm on the sample data. BIRCH [114] summarizes the entire dataset into a CF-tree and then runs a hierarchical clustering algorithm on the CF-tree to get clustering result. In fact, since the size of dataset is reduced with the sampling/summarization techniques, any typical clustering algorithms and cluster validation techniques that have acceptable non-linear computational complexity can be applied. This "sampling/summarization – iterative cluster analysis" framework has been commonly recognized as a practical solution to large-scale cluster analysis.

However, the above two-phase framework does not address the questions for the entire large dataset that are frequently asked by some applications: 1) what is the cluster label for a particular data record which may not be in the representative dataset? 2) what are the data records in the entire dataset that belong to a particular cluster? Therefore, we also need to extend the intermediate clustering result to the entire dataset, which requires the third phase - labeling data on disk with the intermediate clustering result. Previous research on clustering with the three-phase framework has been primarily focused on the first two phases. Surprisingly, very few studies have considered the efficiency and quality of the disk-labeling phase.

Disk-labeling also provides the opportunity to review and correct the errors generated by the

first two phases. For example, sampling/summarization tends to lose the small clusters in the representative dataset. When sampling is applied in the first phase, it is easy to understand that small clusters might be lost for small sample size. This is also true when summarization is done in a high granularity. When a CF-tree in BIRCH, for instance, is relatively small compared to the number of records, a leaf node will possibly cover a large spatial locality and we have to consider all small clusters in the leaf as one cluster. Although many applications only consider the primary clustering structure, i.e., the large clusters, small clusters may become significant for some applications. Thus, there is a need monitoring the small clusters possibly missed by the first phase.

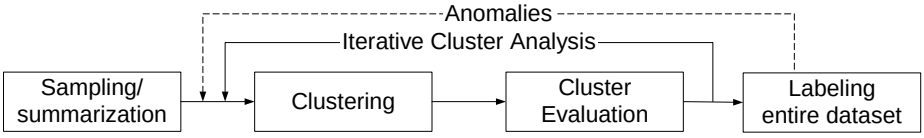

Figure 3: Three phases for cluster analysis of large datasets, (Sampling/summarization – Cluster Analysis – Disk Labeling)

(3) Problems with Irregularly Shaped Clusters. Many automated clustering algorithms work effectively in finding clusters in spherical or elongated shapes but they cannot handle arbitrarily shaped clusters well [53], neither can traditional validation methods, which are primarily statistical methods,effectively validate such clusters [56, 93].

Particularly, in some applications, irregularly shaped clusters may be formed by combining some regular clusters or by splitting one large cluster based on the domain knowledge. Most of the existing clustering algorithms do not allow the user to participate in the clustering process until the clustering algorithm is completed. Thus, it is inconvenient to incorporate the domain knowledge into the cluster analysis, or to allow the user to steer the clustering process that totally employs automated algorithms.

We observe that visualization techniques have played an important role in solving the problem of irregularly shaped clusters in large datasets. Some visualization-based algorithms, such as OPTICS [8], tried to find the arbitrarily shaped clusters, but they are often only applicable to small datasets and few studies have been performed for large datasets. The iVIBRATE approach described in this work fills in this gap with the visualization-based three-phase framework and solves the particular problems related to the integration of the three phases under the framework.

(4)Problems with Disk Labeling. When disk labeling is used as the last phase of clustering large dataset, it assigns a cluster label to each data record on disk based on the intermediate clustering result. Without an effective labeling phase, a large amount of errors can be generated in this process.

In general, the quality of disk-labeling depends on the precise description of cluster boundaries. All existing labeling algorithms are based on very rough cluster descriptions [66], such as a centroid or a set of representative cluster boundary points for a cluster. A typical labeling algorithm assigns each data record on disk to a cluster that has its centroid or its representative boundary points closest to this data record. Centroid-based labeling (CBL) uses the cluster center (centroid) only to represent a cluster; Representative-point-based labeling (RPBL) uses a set of representative points on cluster boundary to describe the cluster. The latter is better because it provides more information about the cluster boundary. With RPBL, the quality of boundary description mainly depends on the number of representative points, which could be very large for some irregular cluster shapes or large clusters. However, it is always not easy for the user to determine the sufficient number of representative points for a particular dataset.

In particular, the cluster boundary cannot be precisely defined with only the sample dataset. Cluster boundaries often continue to evolve as we incorporate more labeled records during the disk labeling phase. We name it the "cluster boundary extension" problem and will describe it in detail later.

3.2 The Scope and Contributions of This Research

We have summarized four key problems in clustering large datasets: 1) the three-phase framework is necessary for reducing the time complexity of an iterative cluster analysis; 2) extending the clustering result on the representative dataset to the entire large dataset can raise significant problems; 3) clustering and validating irregularly shaped clusters in large datasets is important but difficult; and 4) existing disk-labeling algorithms may result in large errors primarily due to the imprecise cluster boundary description.

We also explicitly identify that the problem of cluster boundary extension is a big challenge in the labeling phase if sampling is applied in the first phase. Figure 4 shows the clusters evolving from

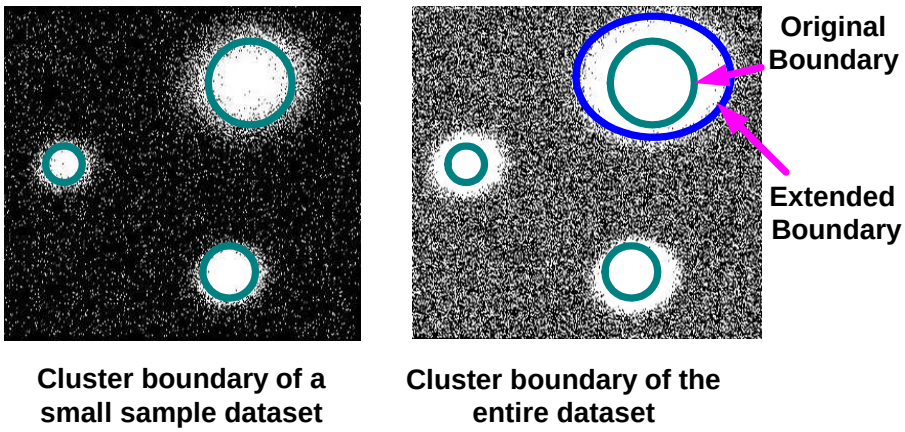

Figure 4: Comparing the cluster boundary of small and large dataset (data points are white)

the small ones in the representative dataset to the larger ones in the entire dataset, where boundary extension results in significant difference in cluster definition. The point density over the initial boundary could increase significantly as the number of labeled records increases, which naturally leads to the boundary extension problem. Especially, when the sample size is much smaller than the size of the large dataset, e.g. $< 1\%$ of the original data records, boundary extension can cause significant errors if labeling algorithms fail to adapt to it.

Boundary extension can also cause two additional problems. 1) For the regular spherical clusters as shown in Figure 4, existing labeling algorithms usually either assign all outliers to the nearby clusters, or treat the cluster members in the extended areas as outliers. As a result, none of them can deal with outliers and boundary extension effectively. For irregular cluster boundary, the situation can be worse. 2) Boundary extension might also result in the overlapping of different clusters, which are originally separated in the representative set. Monitoring boundary extension allows us to recheck and adjust the initial cluster definition.

To address all of the above problems, we propose the iVIBRATE framework – an interactive visualization based three-phase framework for clustering large datasets. The iVIBRATE framework includes the three phases "Sampling – Visual Cluster Analysis – Visualization-based Adaptive Disk-labeling". In this chapter, we introduce the two main components: visual cluster analysis and visualization-based adaptive disk-labeling, while focusing on the important issues in integrating the three components in the iVIBRATE framework. We also demonstrate how to analyze very large

datasets with the iVIBRATE framework.

In the clustering phase, we use visual cluster analysis, including visual clustering and visual cluster validation, to allow the user to participate in the iterative cluster analysis, reducing both the length of single iteration and the number of iterations. We develop a visual cluster rendering system, VISTA, to perform "visual cluster analysis". The VISTA system interactively visualizes multi-dimensional datasets (usually $<$50 dimensions). It allows the user to interactively observe potential clusters in a series of continuously changing visualizations. More importantly, it can incorporate the algorithmic clustering results, and serve as an effective validation and refinement tool for irregularly shaped clusters.

In the disk-labeling phase, we develop the Adaptive ClusterMap Labeling subsystem. ClusterMap encodes the irregular cluster boundaries defined on the visualization that is generated by the VISTA subsystem. The algorithm automatically adapts to the boundary extension phenomenon, clearly distinguishes outliers, continuously detecting irregular shaped clusters, and visually monitoring the anomalies in labeling process. As a result, the Adaptive ClusterMap labeling phase generates fewer errors than the existing disk-labeling algorithms.

When the three phases are integrated in the iVIBRATE framework, errors caused by the improper operations in the prior phases may propagate to the later phases. Visualization helps to detect and reduce such errors. We identify and analyze the issues related to the integration of the phases, and develop the theory and tools to monitor the possible errors. The iVIBRATE framework is evaluated with real and synthetic datasets and the experimental results show that iVIBRATE can take advantage of visualization and user interaction to generate high-quality clustering results for large datasets.

3.3 Related Work

Clustering Large Data. We have described four challenges related to clustering large datasets: time complexity (scalability), sampling/summarization based clustering, irregular clusters and disk-labeling. Although each of these issues has been studied, there is surprisingly little study on how they impact on the cluster quality when sampling/summarization, iterative cluster analysis, and disk labeling are integrated into a unifying framework for exploring the complex cluster structures in

large datasets.

Concretely, time complexity of clustering algorithm has been addressed from early on. K-means algorithm [65] is the most popular algorithm with linear time complexity. Most studies related to K-means assume that the clusters are in spherical shapes. Recently, there are some other algorithms [94, 39, 58] having started looking at the problem of clustering irregular clusters in linear/near-linear time.

Dealing with arbitrarily shaped clusters is well-recognized as a challenging problem in clustering research community. A number of clustering algorithms have aimed at this particular problem, such as CURE [53], CHAMELEON [70], DBSCAN [39], DBCLASD [108], WaveCluster [94], DENCLUE [58] and so on. The semi-automatic algorithm OPTICS [8], derived from the DBSCAN algorithm, shows that visualization can be very helpful in cluster analysis. However, all these algorithms are known to be effective only in low dimensional (typically, $<$10D) datasets or in small/medium datasets with thousands of records.

A general cluster analysis framework is described in a review paper [66], showing that cluster analysis is usually an iterative process. One approach to address the scalability of iterative clustering analysis is the use of the "sampling(summarization) – clustering–labeling" framework, represented by CURE [53] and BIRCH [114]. However, the labeling phase and interactions between the phases have not been sufficiently addressed.

As far as the summarization/sampling phase is concerned, instead of using BIRCH summarization, Bradley et al. [16] suggest using sufficient statistics to model the data summary. In comparison to summarization, sampling is used more extensively in data analysis: commercial vendors of statistical packages (e.g., SAS) typically use uniform sampling to handle large datasets. Vitter's reservoir sampling [104] represents an efficient uniform sampling technique. The main problem with uniform sampling is the loss of small clusters. CURE proposed a method to estimate the minimum sample size if the size of entire large dataset and the smallest size of clusters are known. However, the minimum sample size shall increase dramatically with the increase of dataset size. Thus, for a fixed sample size, it is also necessary to develop methods monitoring the small clusters in order to maintain the consistency in cluster analysis.

Another popular sampling approach is density-biased sampling proposed by Palmer et al. [85].

A density-biased sampling preserves small clusters in the sampling process. However, this technique also skews the actual size of large clusters, introducing too much inconsistency between the clusters in the sample set and the actual clusters in the entire large dataset. It raises particular problems in the later two phases, which are not our focus in this research.

The existing disk labeling solutions heavily depend on the concrete cluster representations generated at the iterative cluster analysis phase. Existing cluster representations can be classified into four categories [66] : centroid-based, boundary-point-based (representative-point-based), classification-tree-based and rule-based representations. The centroid representation is the most popular one. Many clustering algorithms in fact only generate cluster centroids, for example, K-means and most hierarchical algorithms. For boundary-point-based representations, good representative boundary points are often difficult to extract. The most typical algorithm for generating the representative points is CURE [53]. Classification-tree-based and rule-based representations are equivalent (each path in the classification tree can be represented as a rule), however, both are inconvenient to describe high-dimensional data or complicated clustering structure.

Document Clustering in IR and Linkage-based clustering in Network Analysis. Most of the research on clustering large datasets can be roughly categorized into three areas: scientific/business data clustering [65, 66], document clustering [106, 34, 89, 91, 112, 96], and linkage-based clustering for large scale network analysis [66, 54, 84].

In scientific/business data clustering, the data is already formalized as a set of multi-dimensional vectors (i.e., a table). However, in document clustering, the original data is text data. Most document clustering techniques are focused on the two steps before applying clustering algorithms: extracting keywords/constructing the numerical features [111], and defining a suitable similarity measure [10]. Given the vector representation of documents and the similarity measure, document clustering is to some extent similar to scientific/business data clustering. Since document collections have become larger and larger with the wide spread of Internet-based applications, we expect that the iVIBRATE framework can also be extended to clustering large sets of documents.

Graph mining or linkage-based clustering [66] has received a growing interest in the recent

years due to increased interests in analyzing large-scale networks, such as peer to peer online communities [87], and social networks [84]. Linkage-based clustering is also used in clustering categorical datasets [54]. Most of the business/scientific data clustering algorithms utilize the distances between multi-dimensional data points (records) to compute and derive data clusters, while most of the linkage-based clustering algorithms utilize the node connectivity as a main measurement to understand and derive the interesting clustering structure of the network. Thus, linkage-based clustering algorithms aim at finding communities in networks − groups of vertices within which connections are dense but between which connections are sparser.

A commonalty of data clustering, document clustering, and node clustering is the fact that they all emphasize on efficient algorithms to speed up the clustering process of large datasets. However, the subtle differences between distance based measure and connectivity-based measure may influence how the clustering algorithms are devised and what factors are critical to the performance of the algorithms. Due to the scope of this research, we will confine our discussion to the general clustering problem to the business and scientific datasets.

Visualization of Multidimensional Data. Information visualization has demonstrated great advantages in multi-dimensional data analysis. Here we only discuss the scatter-plot-based techniques because they are the most intuitive techniques for cluster visualization. The early research on projection-based general data visualization is the Grand Tour [9]. The Grand Tour tries to find a series of smoothly changed projections that map data to two orthogonal dimensions, so that the user can look at the high-dimensional data from different perspectives. In order to reduce the huge search space for cluster visualization, Projection Pursuit is also used to identify some of interesting projections [30]. Yang [110] utilizes the Grand Tour technique to show projections in an animation. Dhillon, et al. [36] aimed at precise visualization of the clusters, but the technique is only effective for 3 clusters. When more than 3 clusters exist, it requires the help of the Grand Tour technique. The above systems aim at visualizing the datasets in continuously changed projections, which is similar to our VISTA system. However, it is well known that generating continuously changing visualizations in Grand Tour systems often involves complicated computation, and their visualization models are generally not intuitive to users. Most importantly, they do not fully utilize the power

of interaction and heuristic rendering rules. Compared to the Grand Tour models, the VISTA visualization model has several advantages: 1) it provides convenient parameter adjustments; 2) it is enhanced by the heuristic rendering rules and the intuitive interpretation about the rules for finding the satisfactory cluster visualization; 3) continuously changing the VISTA visualization is very easy and fast, either in ARR mode or ADDR mode (see section 3.5.3), which produces the effect of animation at low cost.

Different from the dynamic visualization systems, like the Grand Tour and VISTA, there are static multidimensional visualization techniques such as Scatterplot Matrices, coplots, Parallel Coordinates [64], Glyphs [80], multidimensional stacking [74], and FastMap based visualization [42]. A nice tool, XvmdvTool [105], implements some of the above static visualization techniques. Some techniques are extended to specifically visualize the clustering structures discovered by clustering algorithms, such as IHD [109] and Hierarchical Clustering Explore [92]. However, these techniques are not specifically designed to address the difficult clustering problems: irregularly shaped clusters, domain-specific clustering structure, and problems in labeling clusters in very large datasets. Most of them are also limited by the dimensionality (10-20 dimensions at maximum).

Star Coordinates [68] is a visualization system designed to interactively visualize and analyze clusters. We utilize the form of Star Coordinates and build the α-mapping model in our system. α-mapping model extends the ability of the original mapping used in Star Coordinates, and the mechanism of visual rendering behind this model can be systematically analyzed and understood [22]. RadViz [60] utilizes the same coordinates system with a non-linear mapping function, which makes it difficult to interpret the visual clusters in the generated visualization. HD-Eye [59] is another interesting interactive visual clustering system. It visualizes the density-plot of the interesting projection of any two of the k dimensions. It uses icons to represent the clusters and the relationship between the clusters. However, it is difficult for the user to synthesize all of the interesting 2D projections to find the general pattern of the clusters. In fact, visually determining the cluster distribution solely through user interaction is not necessary. A more practical approach is to incorporate all available clustering information, such as algorithmic clustering results and the domain knowledge, into the visual cluster exploration.

3.4 The iVIBRATE Framework

In this section, we first briefly give the motivation and the design ideas of the iVIBRATE development, and then describe the components and working mechanism of iVIBRATE.

Motivation. In the three-phase framework, cluster analysis involves the ”clustering - analysis/evaluation” iteration, which can be concretely described in the following steps:

1. Run a clustering algorithm with the initial setting of parameters.
2. Analyze the clustering result with statistical measures and the domain knowledge.
3. If the result is not satisfactory, adjust the parameters and re-run the clustering algorithm, or use another algorithm, then go to Step 2 to evaluate the clustering result again until the satisfactory result is obtained.
4. If the result is satisfactory, then perform post-processing, which may include labeling the data records on disk.

Open problems. We first discuss a number of open problems in the above steps, and then we describe how iVIBRATE addresses these problems. Traditional statistical methods, such as variance and intra/inter cluster similarity, are typically used in Step 2, which assume that the shape of cluster structure is hyper-sphere or hyper-ellipse. As a result, these traditional statistical methods have difficulty in effectively validating the irregular cluster shapes [37, 56]. Moreover, with automated algorithms, it is almost impossible to incorporate the domain knowledge. The critical task in step 3 is to learn and determine the appropriate parameter settings. For example, CURE [53] requires the number of representative points and the shrink factor. DBSCAN [39] needs proper ε and *MinPts* to get satisfactory clusters. DENCLUE [58] needs to define the smoothness level and the significance level. These parameter settings are different from dataset to dataset and depend primarily on the user to try different parameters and find the “best” set of parameters by hand. Therefore, there is a need to help the user to easily find the appropriate parameter setting, when automated algorithms are applied. Finally, a coarse labeling algorithm tends to deteriorate the intermediate clustering result as we have discussed.

Bearing these problems in mind, we observed that, if the step 2 and 3 can be carefully combined together, which means that the user can perform evaluation in the course of clustering and be able to refine the clusters at the same time, the length of an iteration would also be greatly reduced. In addition, the user would understand more about the dataset and thus be more confident in their judgment of the clustering results. This motivates us to develop and promote the interactive visual cluster rendering approach.

Cluster Visualization and Visual Validation. Cluster visualization can improve the understanding of the clustering structure. Former studies [73] in the area of visual data exploration support the notion that visual exploration can help in cognition. Visual representations can be very powerful in revealing trends, highlighting outliers, showing clusters, and exposing gaps. According to the paper [95], with the right coding, human pre-attentive perceptual skills enable users to recognize patterns, spot outliers, identify gaps and find clusters in a few hundred milliseconds. In addition, it does not require the knowledge of complex mathematical/statistical algorithms or parameters [72].

Visualization is known to be the most intuitive method for validating clusters, especially clusters in irregular shape. Since the geometry and density features of clusters, which are derived from the distance (similarity) relationship, determine the validity of the clustering results, many clustering algorithms in literature use 2D-plot of clustering results to visually validate their effectiveness on 2D experimental datasets. However, visualizing high-dimensional datasets keeps as a challenging problem.

Static vs. Dynamic Data Visualization. In general, multi-dimensional data visualization can be categorized into static visualization or dynamic visualization. Static visualization displays the data with a fixed set of parameters, while dynamic cluster visualization allows the user to adjust a set of parameters, resulting in a series of continuously changed visualizations. It is commonly believed that static visualization is not sufficient in visualizing clusters [72, 95], and it has been shown that clusters can hardly be satisfactorily preserved in a static visualization [30, 36]. Therefore, we consider using interactive dynamic cluster visualization in the iVIBRATE framework. We observe that a cluster can always be preserved as a point-cloud in visual space through linear mappings. The only problem is that, these point-clouds may overlap with one another and to find certain mapping that can satisfactorily separate the point clouds is mathematically complex. In the iVIBRATE

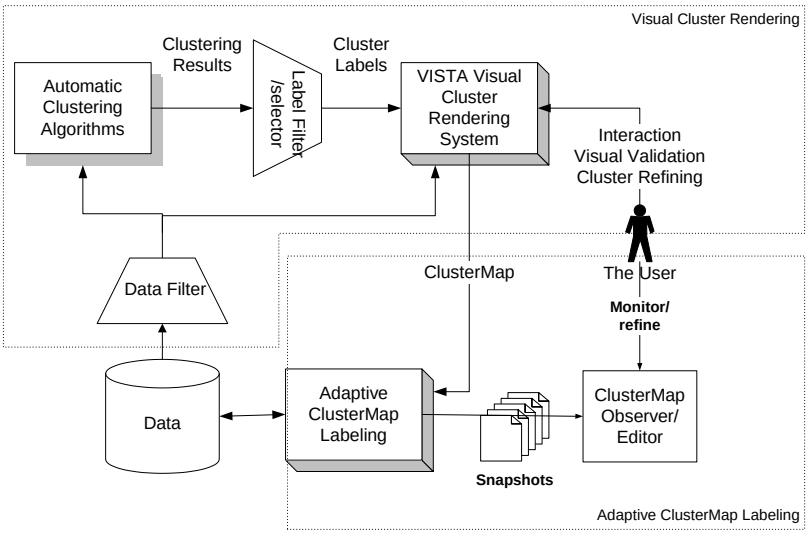

Figure 5: The iVIBRATE framework

framework, we incorporate the combination of visual cluster clues and interactive rendering into the iterative clustering analysis, and refine algorithmic clustering results with heuristic rendering rules, which enables us to identify these point-cloud overlaps quickly and intuitively.

Visualization-based Disk-labeling. Another unique characteristic of iVIBRATE is its visualization-based disk-labeling algorithm. We argue that a fine cluster visualization of a dataset can serve as the visual clustering pattern of this dataset, where the cluster boundary can be precisely described and most outliers can be clearly distinguished. We develop the basic ClusterMap labeling algorithm for obtaining better description of cluster boundary and higher quality of disk-labeling. In order to solve the problem of cluster boundary extension, we also extend the basic ClusterMap algorithm to the Adaptive ClusterMap labeling algorithm.

Components and Working Mechanism. Figure 5 sketches the main components of iVIBRATE framework. We briefly describe each of the main components and the general steps used to analyze the clusters in large datasets.

- **Visual Cluster Rendering** The VISTA system can be used independently to render the clusters in a dataset without incorporating any external information. It can also visualize the result of an automated clustering algorithm or use the result to guide the interactive rendering. By interactively adjusting the parameters, the user can visually validate the algorithmic clustering results through continuously changing visualizations. Using a couple of rendering rules,

which are easy to learn, the user can quickly identify cluster overlaps and improve the cluster visualization as well. In addition, it also allows the user to conveniently incorporate domain knowledge into cluster definition through visual rendering operations. Semi-automated rendering method is also provided for larger number of dimensions ($> 50D$ and $< 100D$).

Data Filter prepares the data for visualization. It handles missing values and normalizes the data. If the dimensionality is too high, dimensionality reduction techniques or feature selection might be applied to get a manageable number of dimensions. When the size of dataset grows past a million items that cannot be easily visualized on a computer display, Data Filter also uniformly samples the data to create a manageable representative dataset. It also extracts certain relevant subsets, for example, one specific cluster, for detailed exploration.

Label Selector selects the clustering result that will be used in visualization as clustering clues or for validation. While a clustering algorithm finishes, it assigns a label to each data record. Label Selector extracts part of the labels corresponding to the data records extracted by Data Filter.

- **Adaptive ClusterMap Labeling** In the iVIBRATE framework, we introduce ClusterMap – a new cluster presentation, and the associated disk-labeling methods. ClusterMap makes the labeling result highly consistent with the intermediate cluster distribution. Adaptive ClusterMap labeling algorithm then automatically adjusts the cluster boundary according to the accumulation of labeled data records at the boundary areas.

  ClusterMap Observer is an interactive monitoring tool. It observes the snapshots, i.e. the changing ClusterMaps during labeling. The snapshots may provide information about the bias of the representative sample set, for example, the missing small clusters, and the anomalies in labeling as shown in section 3.7.2.

A user of iVIBRATE will perform the cluster analysis in the following seven steps. 1) The large dataset is sampled to get a subset in a manageable size (e.g., thousands or tens of thousands records). 2) The sample set is used as an input to the selected automatic clustering algorithms and to

the VISTA visual rendering subsystem. The algorithmic clustering result provides helpful information in the visual cluster rendering process. 3) The user interacts with the VISTA visual cluster rendering system to find the satisfactory visualization, which visualizes the clusters in well-separated areas. Since human vision is very sensitive to the gap between point clouds, which implies the actual boundary of clusters, the interactive rendering works very well in refining vague boundaries or irregular shaped clusters. 4) A ClusterMap is then defined on the satisfactory cluster visualization and used as the initial pattern in ClusterMap labeling. 5) The labeling process will adapt the boundary extension and refine the cluster definition in one pass through the entire dataset. An additional pass might be needed to reorganize the entire dataset for fast processing of queries. During the labeling process, snapshots are saved periodically, which are then used to monitor the anomalies during the labeling process. 6) The user can use the ClusterMap Observer to check the snapshots and refine the extended ClusterMap. 7) To further observe the small clusters that may be omitted in the sampling process, the data filtering component is used to filter out the labeled outliers and performs sampling/visual rendering on the sampled outliers again (for details, see section 3.7.3).

In the following sections, we will introduce the two subsystems, with a focus on the integration of the main components into the framework.

3.5 VISTA Visual Cluster Rendering System

A main challenge in cluster visualization is cluster preservation, i.e., visualizing multi-dimensional datasets in 2D/3D visual space, while preserving the clustering structure. Previous studies have shown that preserving cluster structure precisely in static visualization, if not impossible, is very difficult and computationally expensive [72, 30, 110, 36, 95]. An emerging practical mechanism to address this problem is to allow the user to interactively explore the dataset [72] and to distinguish the visibly inaccurate cluster structure, such as cluster overlapping, broken clusters and false clusters (the situation where the outliers in the original space are mapped to the same visual area and thus form a false visual cluster) through visual interactive operations.

The iVIBRATE visual cluster rendering subsystem (VISTA) is designed to be a dynamic visual cluster exploration system. It uses a visualization model, characterized by the max-min normalization and the α-mapping to produce a linear transformation that maps each multi-dimensional data

point onto a data point in 2D visual space. This mapping model provides a set of visually adjustable parameters, such as the α parameters. By continuously changing one of the parameters, the user can see the dataset from different perspectives. Since the linear mapping does not break the clusters, the clusters in multi-dimensional space are still visualized as dense point clouds (the "visual clusters") in 2D space. And the visible "gaps" between the visual clusters in 2D visual space indicate the *real gaps* between point clouds in the original high dimensional space. However, overlaps between the visual clusters in the 2D space, i.e. the point clouds, may occur with certain parameter settings. We have developed a set of interactive operations and designed several heuristic rendering rules in order to efficiently distinguish the visual cluster overlaps. These developments have shown to be quite effective in achieving desired rendering efficiency.

Since Euclidean distance is the most commonly used distance measure in applications, the current prototype of VISTA subsystem supports clustering with Euclidean distance. For the convenience of presentation, in the rest of the chapter Euclidean distance is used as the default similarity measure. Datasets with other distance measures can be approximately transformed to Euclidean datasets with techniques like multidimensional scaling [32], which will be a part of VISTA extensions in the future work.

3.5.1 The Visualization Model

The VISTA visualization model consists of two linear mappings – max-min normalization followed by α-mapping. For better understanding of the iVIBRATE framework, we briefly introduce the two as follows. Interested readers can refer to the paper [22] for details.

Max-min normalization is used to normalize the columns in the datasets in order to eliminate the dominating effect of large-valued columns. For a column with value bounds [min, max], max-min normalization scales a value v in the column into [-1, 1] as follows:

$$v' = \frac{2(v - \min)}{\max - \min} - 1 \tag{1}$$

where v is the original value and v' is the normalized value.

α-mapping maps k-D points onto the 2D visual space with the convenience of visual parameter tuning. We describe α-mapping as follows. Let a 2D point $Q(x, y)$ represent the image of a k-dimensional (k-D) max-min normalized data point $P(x_1, \ldots x_i, \ldots, x_k)$, $x_i \in [-1, 1]$ in

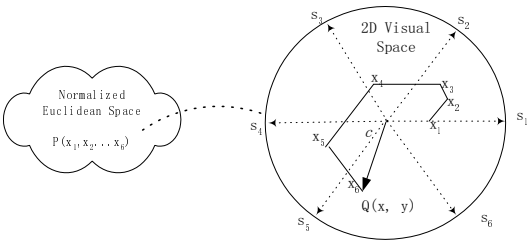

Figure 6: Illustration of α-mapping with $k = 6$

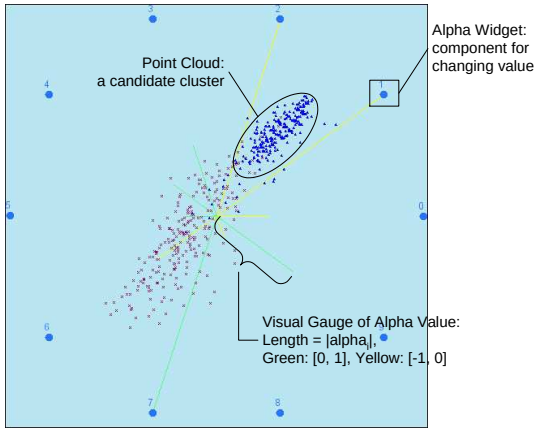

Figure 7: An implementation of α-mapping

2D space. $Q(x, y)$ is determined by the average of the vector sum of the k vectors $\vec{s}_i\dot{x}_i$, where $\vec{s}_i = (\cos(\theta_i), \sin(\theta_i)), i = 1 \ldots k$ and $\theta_i \in [0, 2\pi]$ are the star coordinates [68] that represent the k dimensions in 2D visual space. Formula 2 defines α-mapping.

$$A_{(\theta_1,\ldots,\theta_k)}(x_1, \ldots, x_k, \alpha_1, \ldots, \alpha_k) = (c/k) \sum_{i=1}^{k} \alpha_i x_i \vec{s}_i - \vec{o} \tag{2}$$

i.e. a 2D point $Q(x, y)$ is determined by

$$\{x, y\} = \{(c/k) \sum_{i=1}^{k} \alpha_i x_i \cos(\theta_i) - x_0, (c/k) \sum_{i=1}^{k} \alpha_i x_i \sin(\theta_i) - y_0\} \tag{3}$$

Here, $\alpha_i (i = 1 \ldots k, \alpha_i \in [-1, 1])$ provides the visually adjustable parameters, one for each of the k dimensions. $\alpha_i \in [-1, 1]$ covers a considerable range of mapping functions. Experimental results show that this range combined with the scaling factor c is effective enough for finding satisfactory visualization. θ_i is set to $2i\pi/k$ initially and can be adjusted afterwards. We also proved that adjusting θ values is often equivalent to a pair of α adjustment plus zooming [22]. Thus, it is not necessary to change θ_i in practice. $\vec{o} = (x_0, y_0)$ is the center of the display area.

Lemma 3. *If Euclidean distance is used as the similarity measure, α-mapping preserves the density of dataset, .*

SKETCH PROOF. Let α mapping with $(x_0, y_0) = (0, 0)$ represented by $A(\mathbf{x}) = BC$, where B is the zooming factor $\begin{bmatrix} c & \\ & c \end{bmatrix}$, and C is $1/k \sum_{i=1}^{k} \alpha_i x_i \vec{s}_i$. With the Minkowski inequality and the

Cauchy-Shwartz inequality (see Section 2.1),

$$\begin{aligned} \| C \| &\leq \sum_{i=1}^{k} \frac{|\alpha_i|}{k} |x_i| \| \vec{s}_i \| = \sum_{i=1}^{k} \frac{|\alpha_i|}{k} |x_i| \\ &\leq \sqrt{\sum_{i=1}^{k} \frac{|\alpha_i|^2}{k^2} \sum_{i=1}^{k} x_i^2} \\ &< \sqrt{\sum_{i=1}^{k} x_i^2} \end{aligned}$$

Therefore, C is a contracting projection and α mapping is the combination of a scaling transformation and a contracting projection, which preserves the density of dataset (see Section 2.3). □

Since α-mapping preserves density, there are no "broken clusters" in the visualization, i.e., the visual gaps between the point clouds reflect the real gaps between the clusters in the original high-dimensional space. All we need to do is to separate the possibly overlapped clusters, which can be achieved with the help of dynamic visualization through interactive operations.

The mapping is adjustable by α_i. By tuning α_i continuously, we can see the influence of the i-th dimension to the cluster distribution through a series of smoothly changing visualizations, which usually provides important clustering clues. The dimensions that are important to clustering will cause *significant changes* to the visualization as the corresponding α values are continuously changed.

α-mapping based visualization is implemented in the VISTA subsystem as shown in Figure 7. The coordinates are arranged around the display center and the α-widgets are designed for interactively adjusting each α value. However, the above visual design also limits the number of dimensions that can be visualized and manually manipulated. In the current prototype of VISTA, users can comfortably manually render up to 50 dimensions. Although the system can visualize more than 50 dimensions, we suggest using the semi-automated rendering method instead that will be introduced in section 3.5.3.

Comparison with RadViz visualization model It is worth mentioning that the RadViz system [60] is also based on star coordinates, however, it uses a totally different mapping model. Let $P(x_1, \ldots x_i, \ldots, x_k)$ be the normalized point as above.

$$RV(x_1, \ldots, x_k) = \frac{\sum_{i=1}^{k} x_i \vec{s}_i}{\sum_{i=1}^{k} x_i} - \vec{o} \tag{4}$$

From Equation 4, we can see that RadViz mapping normalizes the contribution of each x_i by all dimensional values. The factor $(\sum_{i=1}^{k} x_i)^{-1}$ renders the mapping as a *non-linear* mapping, which leaves the "visual clusters" difficult to interpret. In fact, the original RadViz visualization is also static – as long as the ordering of k dimensions is determined a unique visualization is generated. Although it is easy to add a set of similar "α" parameters into the model, it depends on further study to develop or understand the potential interactive rendering rules. Since the two mapping models are totally different, our rendering rules and methods presented in the next sections and the paper [22] cannot be easily applied to the RadViz mapping model.

3.5.2 The Rules for Interactive Visual Rendering

To understand the basic visual rendering rules, we should investigate the dynamic properties of the visualization model, especially, the most important interactive operation – α-parameter adjustment (or simply, α-adjustment). α-adjustment changes the parameters defined in Eq. (2). Each change refreshes the visualization in real time (about a couple of hundred milliseconds, depending on different hardware configurations and the size of dataset), generating dynamically changing visualizations. α-adjustment enables the user to find the *dominating dimensions*, to observe the dataset from different perspectives, and to distinguish the real clusters from cluster overlaps in continuously changing visualizations.

Continuous α-parameter adjustment of one dimension reveals the effect of this dimension on the entire visualization. Let $X(x_1, \ldots, x_k)$ and $Y(y_1, \ldots, y_k)$, $x_i, y_i \in [-1, 1]$ represent any two normalized points in k-D space. Let $\|\vec{v}\|$ represent the length of vector $\vec{v}$. We define the *visual distance* between X and Y is:

$$\begin{aligned} vdist(X,Y) &= \|A(x_1,\ldots,x_k,\alpha_1,\ldots,\alpha_i,\ldots,\alpha_k) - A(y_1,\ldots,y_k,\alpha_1,\ldots,\alpha_i,\ldots,\alpha_k)\| \\ &= \|(c/k)\sum_{i=1}^{k}\alpha_i(x_i - y_i)\vec{s}_i\| \end{aligned} \quad (5)$$

which means if x_i and y_i are close, changing α_i does not change the visual distance between X and Y a lot – the dynamic visual effect is that X and Y are moving together when α_i changes. Meanwhile, neighboring points in k-D space also have similar values in each dimension as Euclidean distance is employed. Thus, we can conclude that the neighboring points in k-D space,

which should belong to one cluster, not only are close to each other in 2D space, but also tend to move together in any α-adjustment; while those points that are far away from each other in k-D space may move together in some α-adjustment but definitely not in all α-adjustments. This property makes α-adjustment very effective in revealing the visual cluster overlaps. In addition, point movement can also reveal the value distribution of individual dimension. If we adjust the α value of the dimension i only, the point movement can be represented by:

$$\begin{aligned} \Delta(i) &= A(x_1, \ldots, x_k, \alpha_1, \ldots, \alpha_i, \ldots, \alpha_k) - A(x_1, \ldots, x_k, \alpha_1, \ldots, \alpha_i', \ldots, \alpha_k) \\ &= (c/k)(\alpha_i - \alpha_i')x_i\vec{s_i} \end{aligned} \tag{6}$$

which means that the points having larger x_i will be moving "faster" along the i-th coordinate, and those having the similar x_i moving in a similar way. The initial setting of α values may not reveal the distribution of an individual dimension as Figure 8 shows. However, by looking at the density centers (the moving point cloud) along the i-th axis as α_i changes, we can easily estimate the value distribution along i-th dimension. In Figure 8, we sketch that point movement and point distribution can be interpreted intuitively with each other.

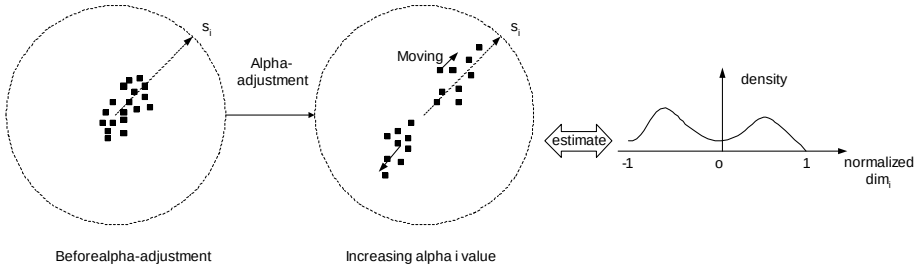

Figure 8: α-adjustment, dimensional data distribution and point movement

In interactive visual rendering, some dimensions show "significant change" on visualization in continuous α-adjustment, i.e., changing its α value results in distinct point clouds moving in different directions, or causes the visible "gaps" between point clouds to emerge. These dimensions play important roles in visual cluster rendering and thus we name them as "visually dominating dimensions", and the others as "the fine-tuning dimensions". The dominating dimensions usually have skewed distributions, where more than one distinctive mode exist on the distribution curve. For example, dimensions with near uniform distribution are definitely not dominating dimensions and dimensions with normal distribution are also less likely to be dominating, however, possibly

useful in refinement of visualization.

Since the main goal of VISTA interactions is to distinguish the possible visual cluster overlaps, we can apply the following rules in visual rendering:

Visual Rendering Rule 1. *Sequentially render each dimension. If the dimension is a visually dominating dimension, increase its α value to certain degree so that the main point clouds are satisfactorily distinguished.*

Visual Rendering Rule 2. *Use the fine-tuning dimensions to polish the visualization. Adjust their α values finely so that the visualization clearly shows the cluster boundaries.*

Guided by the above simple visual rendering rules, a trained user can easily find the satisfactory visualizations. While combined with the cluster labels that are generated by automatic clustering algorithms (for example, K-Means algorithm), the rendering becomes even easier. During the rendering process, we can intuitively validate the algorithmic clustering results and conveniently incorporate the domain knowledge into the clustering process [22], which are difficult for most automated clustering algorithms.

3.5.3 Semi-automated Rendering

When the number of dimensions grows to a considerably large number (> 50 and < 100 dimensions), manually rendering the dimensions becomes a difficult job. In the VISTA subsystem, we provide a semi-automated rendering method to automate the rendering of this type of datasets. Together with the visual rendering rules we have presented, the semi-automated rendering method can be quite efficient.

Concretely, our semi-automated rendering is performed in two stages: automatic random rendering (ARR) followed by automatic dimension-by-dimension rendering (ADDR). A simple version of random rendering is defined as follows. Let a dataset S have d dimensions and N records. Each dimension i ($1 \leq i \leq k$) is associated with an initial α value, say α_i. Random rendering can be done in any number of rounds until some rough pattern of cluster distribution is observed. In each round, the α_i value is changed by a small constant amount ϵ, ($0 < \epsilon < 1$), but the direction (increase or decrease) is randomly chosen for each α_i. Since the α values are bounded by 1 and -1, the change

is "bounced" back at the ends. By changing α values in this way, rather than randomly assigning them in each round, we can observe that the visualization is more smoothly changed. This type of continuity between the nearby visualizations is important to the user, since the user's reaction might be slower than the change of visualization. When a nice rough pattern is observed, a few successive visualizations will be similar to the observed one, allowing the user to stop ARR around the satisfactory pattern.

After a rough pattern is observed in random rendering, we switch the automated rendering from ARR to ADDR for further refinement. In ADDR, for each dimension i, α_i is continuously changed between [-1,1] by steps. Namely, α_i increases by ϵ at each step from -1 to 1, and decreases by $-\epsilon$ from 1 to -1. When a more refined cluster visualization is accepted by the user, ADDR for the dimension i is stopped and moved to the next dimension. ARR helps to quickly find some sketch of

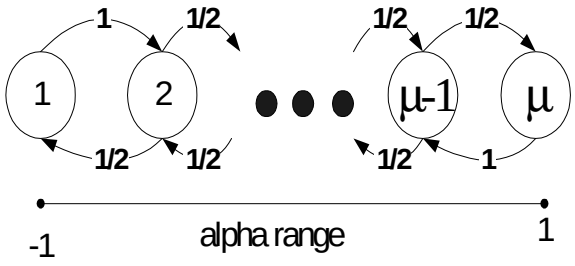

Figure 9: Markov model of random rendering of one dimension.

the cluster distribution and ADDR refines the sketch to get the final visualization. Essentially, the first stage provides the main saving of time with respect to the number of interactions required to find a satisfactory sketch of the cluster distribution pattern, and is the dominating factor in determining how efficient the entire rendering will be. Therefore, we below focus on analyzing and improving the performance of the first stage.

Without loss of generality, we simplify the model as follows: each increase/decrease will move the α to certain fixed points, which is solely determined by the value ϵ. For example, if $\epsilon = 0.2$, the serial of points would be $\{$-1, -0.8, -0.6, ..., 0.8, 1$\}$. Suppose that there are μ such points, including the two endpoints -1, and 1. For simplicity, we call this set of points the μ set of points or μ points for short.

Let P_j, $1 \leq j \leq \mu$ be the probability of setting α to be one of the μ points between [-1,1]. We

can model the ARR process with a Markov chain (Figure 9). It follows that $2P_1 = P_2 = \ldots = P_{\mu-1} = 2P_\mu$ [88], which implies that ARR almost uniformly sets α_i to all values in [-1, 1] (except the two endpoints, which have lower probability).

Now we define the α setting of the satisfactory sketch visualization. Suppose that a sketch of cluster distribution can be observed with the set of α_i value ranges, i.e., as long as the α_i value within the corresponding range, a satisfactory cluster visualization will be observed. We model such a subrange for α_i with λ_i = [λ_{i1}, λ_{i2}], and $|\lambda_i|$ as the number of the μ points that fall into the range λ_i. Therefore, the probability that one ARR operation finds the satisfactory sketch can be estimated by the equation 7.

$$P = \frac{|\lambda_1|}{\mu} \cdot \frac{|\lambda_2|}{\mu} \cdots \frac{|\lambda_k|}{\mu} = \frac{\Pi_{i=1}^{k} |\lambda_i|}{\mu^k} \tag{7}$$

The above equation implies two important factors in terms of the efficiency of ARR. First, the number of effective dimensions, d, is in fact less than k and may vary from dataset to dataset. As the rendering rule 1 suggests, only the "dominating dimensions" are significant to rendering. In other words, the $|\lambda_i|/\mu$ for the minor dimensions can be approximately treated as 1. Second, the individual coverage rate $|\lambda_i|/\mu$ can be increased by reducing the effective α ranges. Based on the analysis of α-adjustment (recall Section 4), smaller α_i values tend to hide the distribution detail over the dimension i, good for polishing, but the larger α values help to distinguish visual cluster overlapping. Thus, in the ARR stage, we can choose to let ARR focus on the reduced ranges, say $[-1, -\beta]$ and $[\beta, 1]$ where $0 < \beta < 1$, for the dominating dimensions.

As observed in experiments, the rate of effective subrange λ_i to the reduced range is often quite large, and there are likely more than one $\Lambda = (\lambda_1, \ldots, \lambda_k)$ range combinations that can visualize the sketch of cluster distribution. Therefore, combined with the rendering rules, it is quite efficient to use ARR as the first step in rendering very high dimensional datasets.

However, ARR is not sufficient to find a detailed cluster visualization. A detailed cluster visualization might confine λ_is to much smaller subranges, which requires the second stage, ADDR, to refine the sketch visualization obtained by ARR. Our experiments show that by using the combination of ARR and ADDR, the cluster visualization of census dataset (68 dimensions) can be captured in around 10 minutes.

3.6 ClusterMap Labeling

In the labeling phase of iVIBRATE framework, we use the adaptive ClusterMap labeling algorithm to effectively extend the intermediate clustering results to the entire large dataset. The concepts of the ClusterMap and the extended ClusterMap are discussed in the paper [21]. Thus, we only provide an overview of the ClusterMap design in this thesis. We refer the readers to the paper [21] for further details.

3.6.1 Encoding and Labeling Clusters with ClusterMap

ClusterMap is a convenient cluster representation derived from the VISTA cluster rendering subsystem. When visual cluster rendering produces satisfactory visualization, we can set the boundaries of a cluster by drawing a visual boundary to enclose it. Each cluster is assigned with a unique cluster identifier. After the cluster regions are marked, the entire display area can be saved (represented) as a 2D byte array (Figure 10). Each cell in the 2D array is labeled by an identifier $-$ a cluster ID ($>$0) if it is within cluster region, or the outlier ID (=0), otherwise. Since the size of array is restricted by the screen size, we do not need a lot of space to save it. For example, the display area is only about 688*688 pixels on 1024*768 screen, slightly larger for a higher resolution, but always bounded by a few mega pixels. As shown in Figure 10, the Cluster Map array is often a sparse matrix, which can also be stored more space-efficiently if necessary. Figure 11 is a visually defined ClusterMap of the 4D "iris" dataset. The boundaries of cluster C1, C2 and C3 were defined interactively.

In addition to the 2D array, we need also to save the following mapping parameters in Table 1 for the labeling purpose.

Table 1: Mapping parameters for ClusterMap representation

$Cmax_j, Cmin_j$	The max-min bounds of each column, $j = 1 \ldots k$
(x_0, y_0)	The center of the visualization
α_j	The k α parameters, $j = 1 \ldots k$
θ_j	The k θ parameters, $j = 1 \ldots k$
c	The scaling factor

ClusterMap representation has several advantages. First, in most situations, ClusterMap provides more details than the centroid-based or representative-point-based cluster representation. Thus,

it is possible to better preserve the intermediate clustering results in the labeling phase. Second, cluster boundaries can be conveniently adjusted to adapt to any special situations or to incorporate domain knowledge as we did in the VISTA system. Finally, with ClusterMap the outliers can be better distinguished. We shall see later that ClusterMap can also be used to conveniently adapt the extension of cluster boundary.

ClusterMap representation can be applied directly in the *basic ClusterMap labeling*. It works as follows. After the ClusterMap representation is loaded into the memory, each item in the entire large dataset is scanned and mapped onto one ClusterMap cell. The mapping follows the same mapping model used in the visual rendering system. Suppose that the raw large dataset is stored on disk in form of N-row by k-column table. We rewrite the mapping formulas as follows:

$$\text{Normalization: } x'_{ij} = \omega_j * (x_{ij} - Cmin_j) - 1 \tag{8}$$

$$\omega_j = 2/(Cmaxj - Cminj)$$

$$\alpha\text{-mapping: } x_i = \sum_{j=1}^{k} \psi_x(j) x'_{ij} - x_0, y_i = \sum_{j=1}^{k} \psi_x(j) x'_{ij} - y_0 \tag{9}$$

where $\psi_x(j) = c\alpha_j \cos(\theta_j)/k$, $\psi_x(j) = c\alpha_j \sin(\theta_j)/k$ and ω_j can be pre-computed, and other parameters, such as c and θ_j are the same as defined in VISTA visualization model.

Sequentially, the algorithm reads the i-th item $(x_{i1} \ldots x_{ik})$ from the k-D raw dataset, normalizes it and maps it to a 2D cell coordinate (x_i, y_i). From the cell (x_i, y_i), we can find a cluster ID label, or a outlier label, depending on the ClusterMap definition.

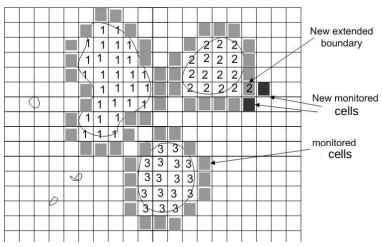

Figure 10: ClusterMap with the monitored area where $\epsilon = 1$

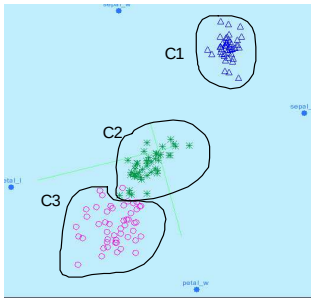

Figure 11: ClusterMap of the 4D "iris" data

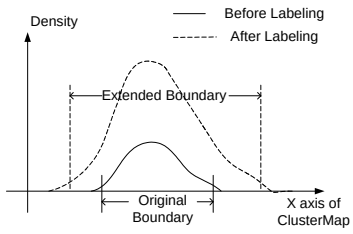

Figure 12: Boundary extension – the cross section of a typical evolving cluster in ClusterMap.

3.6.2 Adaptive ClusterMap for Boundary Extension

In the basic ClusterMap labeling, we assume that the cluster boundary defined on the sample set will not change significantly during the labeling process. However, with the increase of labeled items, the density in the original boundary areas will increase as well. Thus, the original boundary defined on sample set shall be extended to some extent. An example has been shown in Section 1.2 (Figure 4). Boundary extension encloses the nearby outliers into the clusters and may require the mergence of the two nearby clusters if they become overlapped. Figure 12 sketches the possible extension in ClusterMap with the attending of labeled items, in terms of the point density.

Boundary extension is maintained by monitoring the point density around the boundary area. We have the initial boundary defined in ClusterMap representation. We name the cells within the cluster boundary as the "cluster cells" and the cells around the boundary area as the "boundary cells". The initial boundary cells are precisely defined as within a short distance ε away from the initial boundary. All non-cluster cells are "outlier cells" including the boundary cells. We define the density of a cell on the map as the number of items being mapped to this cell. Apparently, the density of boundary cells should be monitored in order to make decision on boundary extension. A threshold density, δ, is defined as two times of the average density of outlier cells. If the density of a boundary cell grows to δ with the attending of labeled items, the boundary cell is turned into a cluster cell and results in the extension of boundary $-$ The non-cluster cells within the ε-distance from the old boundary cell become the new boundary cells. Since the boundary is on the 2D cells, we can use cell as the basic distance unit and the "city block" distance [97] as the distance function to define the ε-distance. ε is often a small number, for example, 1 or 2 blocks from the current boundary. boundary extension keeps consistent with the density of non-cluster area.

To support the above adaptive algorithm, we need to extend the basic structure of ClusterMap. First, for each cell, we need one more field to indicate whether it is a monitored non-cluster cell or not. We also need to keep track of the number of points falling onto each cell. This information is saved at a "Density Map". In addition, δ should be periodically updated according to the average noise level, since the average noise level will also rise with the increase of labeled items. For detailed discussion of the ClusterMap algorithm, please refer to the paper [21]

The Adaptive ClusterMap labeling algorithm can be performed in two scans: The first scan generates an extended ClusterMap and the second scan can be performed to build up a R-tree index on the map for efficient access to the items on disk. After the first scan, the adjusted ClusterMap can also be checked with the ClusterMap Observer to identify the anomalies (section 3.7.2). The second scan is very helpful for many clustering applications that involve similarity search [77] and cluster-based indexing, which require efficient access of the cluster members.

3.6.3 Complexity of Labeling Algorithms

The two key factors that measure the effectiveness of the labeling algorithms are *accuracy* and *computational cost*. Although ClusterMap brings apparent advantages in describing precise cluster boundaries, accuracy will be further evaluated in experiments. We analyze the other important factor: the computational cost in this section for the four labeling algorithms: CBL, RPBL, Basic ClusterMap, and Adaptive ClusterMap.

One way to estimate the cost of a labeling algorithm is to count the number of necessary multiplications. For example, one k-D Euclidean distance calculation costs $O(k)$. Based on the formulas 8 and 9 given in section 3.6.1, we can roughly estimate the cost of the basic ClusterMap labeling. Map reading and parameter reading cost little constant time due to the limited small map size. For each item in the dataset, max-min normalization costs $O(k)$ as shown by Formula 8. α-mapping function costs $O(k)$ to calculate the x and y coordinates with Formula 9, respectively. Locating the cell in ClusterMap to get the corresponding cluster ID costs constant time. Hence, the total cost for the entire dataset is $O(3kN)$, where N is the number of rows in the dataset. While the Adaptive ClusterMap runs with the two scans, the cost is roughly two times of the Basic ClusterMap labeling, i.e., $O(6kN)$.

When kd-tree [45], or other multi-dimensional trees, is used to organize the representative points or centroids, we get the near-optimal complexity for the distance-comparison based labeling algorithms. Let r be the number of clusters and m be the number of representation points per cluster. The cost to find the nearest neighbor point in kd-tree is at least $\log_2(rm)$ distance calculation for RPBL or $\log_2(r)$ for CBL. For a typical RPBL as reported in the CURE paper, the number of representative points has to be greater than 10 $(m \geqslant 10)$ in order to roughly describe the regular

Table 2: Cost estimation of the four algorithms

Algorithm	Complexity	LDS data (N=1M, k=5, r =5)	Census data (N=1M, k=68, r=3)
CBL	[$kN\log_2(r)$, rkN]	4.67	56
RPBL	[$kN\log_2(rm)$, rmkN]	6.5	65
Basic ClusterMap	3kN	4.42	54
Adaptive ClusterMap	$\sim$ 6kN	8.69	108

non-spherical cluster shapes (mainly, the elongated shapes). The number should increase substantially if the irregular cluster shapes are detected. Conservatively, the cost of RPBL will be at least $4kN$, normally a little higher than that of the basic ClusterMap. The cost of CBL should be around $\log_2(r)kN$ or rkN if r is small and a tree structure will increase the cost.

Both CBL and RPBL need a small amount of memory, $O(rk)$ and $O(rmk)$, respectively. Let w be the width and h be the height of the 2D ClusterMap, the basic ClusterMap will need $O(wh)$ memory, which counts for several megabytes in practice. Correspondingly, the adaptive ClusterMap needs about $O(2wh)$ memory.

Table 2 summarizes the formal analysis. The cost on two large datasets, LDS and Census data, which will appear in the later sections, are also listed in Table 2 to give a feeling of the real cost. For both datasets, m is set to 20 and the time unit is second.

In summary, the Adaptive ClusterMap labeling algorithm uses a little more time and space to label the datasets but this small extra cost can bring huge benefits as we will show in the following sections.

3.7 Integrating the Three Phases

Integrating the three phases (Sampling, Visual Cluster Analysis, and Adaptive ClusterMap Labeling) under the iVIBRATE framework presents some interesting and unique challenges. Since the phases are interconnected in sequence, without proper operations in the earlier phases, errors could be propagated and aggravated in the later phases. In this section, we investigate two important issues in integrating the three phases. First, we study the effect of the sampling phase on the later two phases, primarily the impact on determining the max-min bounds from samples (section 3.7.1) and exploring the small clusters hiding in outliers (section 3.7.3). Second, we analyze the possible

influence of Visual Cluster Analysis on the labeling phase, and develop some anomaly monitoring methods to control and reduce the errors (section 3.7.2).

3.7.1 Determine the Max-min Bounds from Samples

The VISTA subsystem requires to determine the max-min bounds for normalization, denoted by "$Cmax_j$" and "$Cmin_j$". These bounds are used not only in the rendering phase by the α function but also in the labeling phase by the ClusterMap algorithms. Thus, these bounds should be kept unchanged throughout the three-phase clustering process. Max-min normalization is the first step in the VISTA visualization model (Section 3.5.1), which prepares the data for α-mapping without loss of any information for visual cluster rendering. However, since the max-min bounds are obtained from the sample set, they may differ from the actual bounds for the entire dataset. Inappropriate setting of bounds may cause additional errors in both the clustering phase and the labeling phase.

Concretely, the effect of inappropriate bounds is twofold. First, if the max and the min bounds are too tight (i.e., the two bounds are too close to one another), even though they enclose all samples, there might be high out-of-bounds rates for the entire large dataset, which increases the amount of errors generated at the labeling phase. On the other hand, if the max-min bounds are too loose, most values are scaled down to a narrow range and the difference between the values cannot be observed efficiently in cluster rendering. Recall that the α-mapping in equation 2 of Section 4.1 shows two possible ways that we can adjust the visual parameters in order to observe the visual difference between different values: one is to adjust the α_i values ($1 \leq i \leq k$) and the other is to alter the scaling factor c. Since the α values ($\alpha_1, \ldots, \alpha_k$) are restricted in the range of [-1, 1] for the purpose of efficient interactive rendering, we might have to adjust the scaling factor c to a large value, which, however, could improperly enlarge the entire visualization and leave some part of visualization out of the display area. Therefore, the ideal bound setting will be located at a narrow range.

The first problem is how large the sample bounds can work approximately as the overall bounds. We address the problem by studying the relationship between the sample value bounds and the sample size $-$ if we just use the sample value bounds as the overall max-min bounds, how many sample points do we need in order to find the bounds that are also appropriate for the entire dataset, i.e., enclosing almost all points? The problem of bounds estimation based on the sample data can

be formalized as follows.

Let n denote the size of the sample dataset and p denote the probability of points in the entire dataset enclosed by the sample bounds. Since bounds estimation for each column is independent, without loss of generality, we can treat the values from one column as samples of a random variable X. We now estimate the bounds for the random variable X with the sample set, so that the bounds cover $100p$ percent of the distribution of X with certain confidence level. This problem can be exactly modeled as *Tolerance Interval* [29].

Definition 10. *A tolerance interval* $(r \leqslant X \leqslant s)$ *with tolerance coefficient* γ *is a random interval. Its range* $[r, s]$ *includes at lease* $100p$ *percent of distribution with the probability* γ.

In our case, we fix the two end points as the two order statistics, $X_{(1)}$ and $X_{(n)}$, i.e. the max and min values of the sample set, for easy processing. The above definition then can be rephrased as:

$$P[P(X_{(1)} < X < X_{(n)}) \geqslant p] = \gamma \tag{10}$$

Let $F_X(x)$ be the distribution function of X and $U_{(n)}$ and $U_{(1)}$ be the max and min values of n uniform samples in [0, 1]. $P(X_{(1)} < X < X_{(n)})$ is equal to $F_X(X_{(n)}) - F_X(X_{(n)}) = U_{(n)} - U_{(1)}$. Therefore, without knowing the distribution of X, we can find the distribution of $U_{(n)} - U_{(1)}$ instead, which is solely related to the order statistics of uniform distribution. Let $U = U_{(n)} - U_{(1)}$, it is easy to find the joint distribution with order statistics. We can get the density distribution of $U = U_{(n)} - U_{(1)}$, $f_U(u, v)$ as follows.

$$f_U(u, v) = n(n-1)u^{n-2}(1-u) \tag{11}$$

Now, since $\gamma = P(U \geqslant p)$, we can get the following relation between γ, p, and n.

$$\gamma = \int_p^1 n(n-1)u^{n-2}(1-u)du \tag{12}$$

The right side of the equation is the *incomplete Beta function* (represented as $betainc(1 - p, 2, n - 1)$ in Matlab). Fixing one of the three parameters γ, p, and n, we can infer the relation between the other two parameters. We are more interested in the range of the sample size for a large p so that the sample bounds can cover almost all points in the entire dataset. By setting p to a very high probability, 0.999, we find the relationship between γ and sample size n as Figure 13 shows.

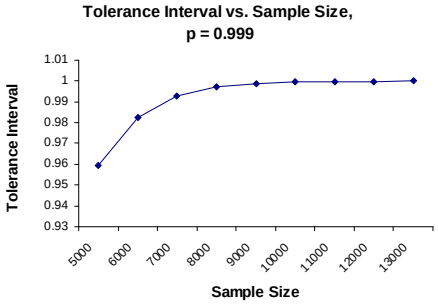

Figure 13: The relation between γ and n, $p = 0.999$

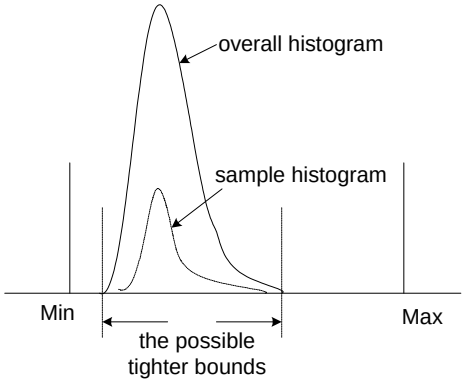

Figure 14: The possible tighter bounds for skewed distribution

At sample size $n \approx 13,000$, the tolerance level almost reaches 100%, which means that we can confidently say that the max-min value bounds of a sample set in size of $n = 13,000$ or larger are the bounds which cover 99.9% records in the entire dataset. Note that the number 13,000 is induced without any assumption about the data distribution and the sample size for real datasets. For real datasets that have some special distributions, the sample size should be smaller as shown in section 3.3. Therefore, we consider this as the upper bound of sampling size.

Our experiments with the first prototype of iVIBRATE shows that its VISTA cluster rendering subsystem can comfortably handle up to 50,000 items with 30-50 dimensions in near real time, in a common computer system environment (for example, CPU 1.5Ghz, memory 256M) [22]. Thus, the VISTA subsystem can also comfortably render a large sample set ($n \approx 13,000$), which definitely contains the max-min bounds for the entire dataset.

The second problem is that the initial max-min bounds based on the sample value bounds might also be too wide, when the distribution is skewed as shown in Figure 14. In a skewed distribution, almost all points are located within a narrow range, with small amount of points far away from the center. This can frequently happen in most real datasets. In this case, if we simply use the sample bounds for normalization, the cluster rendering subsystem may not work efficiently as we have discussed. This can be checked by the histogram of the sample data column. If the skews are found in the sample dataset, we may need to check the histogram for the entire dataset to carefully narrow down the bounds. Based on the above analysis, we suggest the following steps to choose the normalization bounds for each column.

1. Sample the dataset to get a sample set in size around 13,000;

2. Find the max-min bounds of the sample set as the initial bounds for each column and build the histograms for the columns with the sample set;

3. If some columns have very skewed distribution with a few outliers, we build the histograms for these columns from the entire dataset. The loose bounds can be narrowed down according to the histograms for entire dataset, while maintaining the out-of-bounds rate as $1 - p$, e.g., p=0.999.

By doing this, we can avoid the relatively expensive third step for some datasets. However, in the worst case, the overall cost of finding the proper bounds is still quite acceptable. If no skew is found in the second step, the total cost is $O(n)$, where n is the sample size. Otherwise, it is $O(N)$, where N is the size of the large dataset. Since this is a one-time process, $O(N)$ is still not bad.

3.7.2 Monitoring the Anomalies in ClusterMap Labeling

Boundary extension can behave abnormally due to low sample rate, imprecise rendering result, or inappropriate setting of the initial cluster boundary. We first discuss two possible anomalies in the ClusterMap labeling process, and then introduce the methods to monitoring and handling these anomalies.

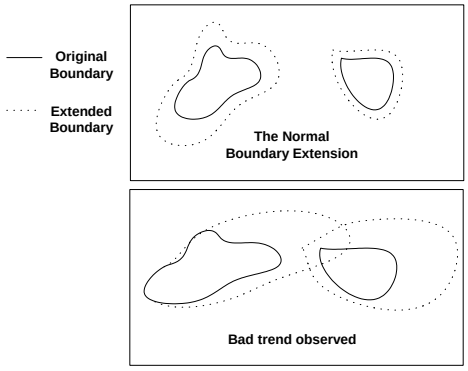

Figure 15: Anomalies that require fine adjustment of α values

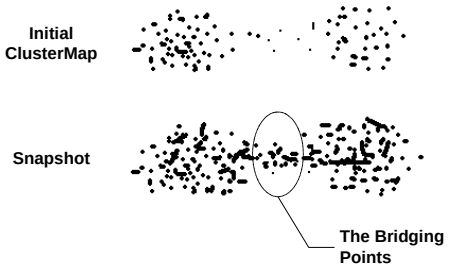

Figure 16: The bridging points

- The first anomaly is the vague cluster boundary. The cluster boundary becomes vague soon after labeling certain amount points, while the normal boundary extension should be slow and

happen uniformly around the boundary throughout the entire labeling process. There are two situations that can cause such a anomaly, illustrated by Figures 15 and 16. First, the initial distances between the clusters are not defined appropriately due to the sample size or lack of visual refinement, which requires the user to tune the initial ClusterMap, e.g., adjusting the α parameters slightly in VISTA. This is illustrated by Figure 15. Second, the other situation is the "bridging points" between the clusters, which are not dense enough in the sample set but they may form the "bridge" that connects the clusters later in the labeling process, as shown in Figure 16. The user has to make decision based on the domain knowledge to either split or merge them.

- The second anomaly is that the ordering of data records on disk may affect the boundary extension. For example, a sequence of data is mapped to a focused boundary area at early stage of labeling thus the a false boundary extension occurs. However, later on no more points are mapped to that area. As a result, this area is falsely extended as a part of cluster. We observe that this error only happens in the situation where such particular data records are stored together and the labeling is done according to the original ordering of data records on disk. This anomaly can be avoided by accessing the data records in a perturbed sequence. We use a method named "sequence perturbation" (Figure 17). To put it simply, if the large data file is regarded as a block file, we equally divide the dataset into s sequences of blocks. In each processing window, we read some data blocks at the head of each sequence and perturb the ordering of the records in these blocks. This can almost eliminate the risk of non-uniformity in data ordering.

In general, these anomalies can be monitored with the "snapshots" of labeling, which are visualized by the tool "ClusterMap Observer". Snapshots are a series of evolving ClusterMaps and density maps, which incorporate the boundary extension and are saved at some time interval during the first scan of Adaptive ClusterMap labeling. The user can observe the snapshots with ClusterMap Observer. If the anomalies are observed, the user can terminate the labeling process early and returns to VISTA subsystem to adjust the original ClusterMap. Figure 18 shows a snapshot with 10 million records labeled. The noisy areas around cluster A and B are not labeled as cluster cells

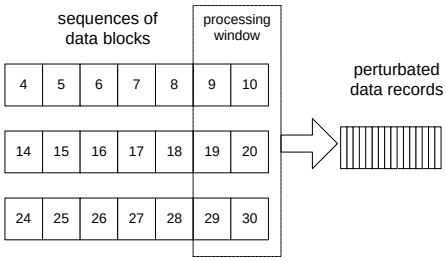

Figure 17: Record perturbation

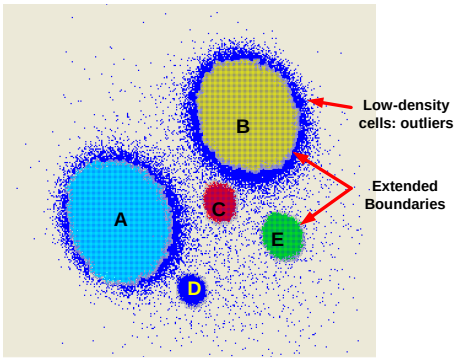

Figure 18: A snapshot of labeling a large dataset, visualized with ClusterMap Observer

since the density of these cells does not reach the threshold. Whether these cells should be included into clusters or not, may depend on the user's requirement. However, the extended boundary can always be edited with ClusterMap Observer, which makes the entire labeling process very flexible and manageable.

3.7.3 Detect and Explore the Small Clusters

Missing small clusters is most likely caused by low sample rate, e.g. less than 1% of the entire dataset. The small clusters may start to emerge as the labeling proceeds, which could be detected in the snapshots of labeling.

If there are small clusters emerging, we can use the following filtering method to confirm and explore the small clusters in detail. In the labeling phase, we run the Adaptive ClusterMap labeling to label all records, and then extract the outliers *only* from the large dataset for visual rendering. If the outlier dataset is still large, it is sampled and rendered in VISTA cluster rendering system again. Since the size of the outlier dataset is usually much smaller than that of the original dataset, one additional sampling for the outlier dataset is often sufficient to discover the small clusters in it. Similarly, the observed small clusters are marked in an *additional* ClusterMap. We can repeat this process until the size of the outliers becomes negligibly small. This process might result in a couple of additional ClusterMaps representing the small clusters at different detail levels. These ClusterMaps are used together to effectively label the interested small clusters.

In this process, the user can always control the "drill-down" level and the size of interested

small clusters. More flexibly, the user can select any interested area of ClusterMap and zoom in to observe the possible small clusters in the corresponding portion of data only. This can be done iteratively, which results in a general extended iVIBRATE framework for hierarchically exploring the clusters in very large datasets (Figure 19). In short, under the iVIBRATE framework, users have more flexibility in monitoring and exploring the details in clustering structure. To our knowledge, no one of the existing approaches has provided such flexibility.

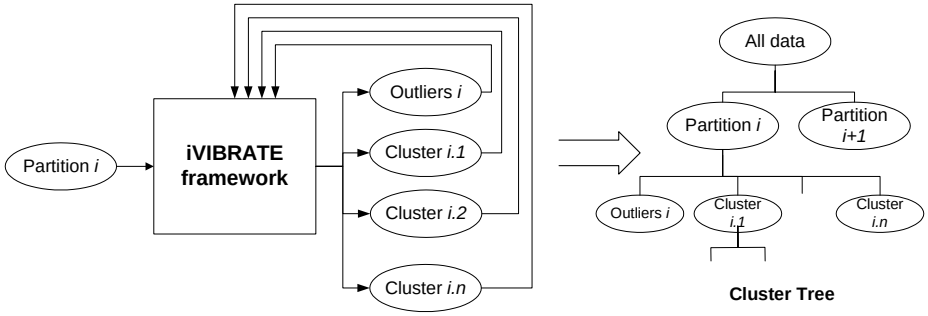

Figure 19: Iterative exploration in the extended iVIBRATE framework

3.8 Experiments

This section presents three sets of experiments. The first set of experiments shows the effectiveness of VISTA visual clustering rendering in finding irregularly shaped clusters. The second set of experiments shows that ClusterMap labeling can handle outliers and irregularly shaped clusters with low computational cost. The third set of experiments demonstrates the advantage of Adaptive ClusterMap labeling. The results show that this visualization powered framework iVIBRATE is more reliable and flexible than any existing approaches.

3.8.1 Datasets and Experiment Setup

The first set of experiments are conducted on a number of well-known datasets that can be found in UCI machine learning database [1]. These datasets, although small or median in size, have irregular cluster distribution, which is an important factor for testing the effectiveness of the VISTA system. We carefully choose these datasets with the following three factors in mind: 1) the current version of VISTA system only concerns the datasets having a manageable number of numerical attributes;

[1] http://www.ics.uci.edu/∼mlearn/Machine-Learning.html

2) clusters in most of the datasets are not in regular spherical shape, the size of cluster may vary greatly, and the distance between clusters can be so close that the algorithmic approaches can easily fail to distinguish; 3) the existing class labels can effectively indicate the irregular clusters. For easy comparison, we also ignore the tiny clusters in some datasets, for example, in "ecoli[2]" data and "shuttle" data.

Dataset	N	# of dim.	# of clusters
Breast-w	699	10	2
Crx	690	15	2
Ecoli	336	7	8
Hepatitis	155	19	2
Ionosphere	351	34	2
Iris	151	4	3
Wine	178	12	3
Shuttle.test	14500	9	7

Table 3: The datasets used in visual rendering.

Two datasets are used for the second set of the experiments. One is the simulated dataset DS1 used in CURE [53]. DS1 is a 2D dataset having five regular clusters, including three spherical clusters, two connected elliptic clusters, and many outliers. In our experiments, DS1 is used to evaluate the effect of outliers on the labeling algorithms. The second dataset is the "shuttle" dataset (STATLOG version, test dataset) introduced in the first set of experiments. It is a 9-dimensional dataset with very irregular cluster distribution. There are seven clusters in this dataset, among which one is very large with approximately 80% of data items, and two are moderately large with approximately 15% and 5% of data items, respectively. Others are tiny clusters and thus ignored in comparison. "Shuttle" dataset is used to evaluate the effect of irregular clusters on the labeling process. These two datasets should show how ClusterMap avoids the common problems of the traditional labeling algorithms.

In the third set of experiments, a simulated 5-dimension large dataset LDS with one million records is designed to test the performance of Adaptive ClusterMap on very large datasets. Figure 28 shows a 10K sample set visualized with VISTA system. LDS simulates 5 clusters – three are approximately spherical, and the other two are in irregular shape. There are also about 1% outliers. LDS is well-designed so that we can approximately predefine the control labels for the entire dataset

[2]There are totally 8 attributes in Ecoli data, but one is the name of E.Coli, which is discarded in clustering.

with small errors. This dataset is used to evaluate the effect of all three factors: outliers, irregular clusters, and boundary extension.

The three labeling algorithms, CBL, RPBL, and ClusterMap are implemented in C++. RPBL is based on the boundary points generated by the CURE clustering algorithm, which was known as a fine RPBL adapted for non-spherical cluster. We run CURE clustering to get the boundary points with the following parameters: the number of representative points is 20 and α (the shrink factor) is set to 0.5 as suggested. We also use ANN (Approximate Nearest Neighbor) C++ library from University of Maryland at College Park to construct kd-trees for RPBL and CBL in order to improve the performance of nearest neighbor search.

3.8.2 Visual Cluster Rendering

In this section we will introduce the experimental result concerning the power of visual cluster rendering system in finding clusters. The VISTA visual clustering system was implemented in Java [3].

When we finish interactive cluster rendering, we mark the cluster areas, in which the points are respectively labeled with the cluster ID. With the original labels in the datasets, we can define the items that are wrongly clustered as the errors, the number of which divided by the size of the dataset is the error rate of visual cluster rendering result.

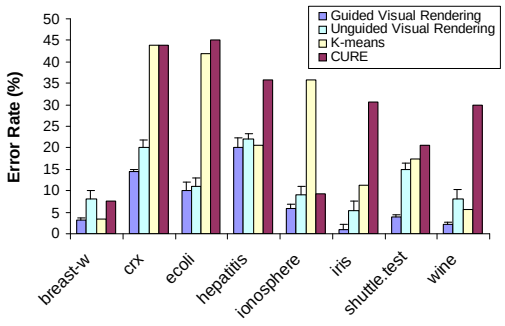

Figure 20: Comparison of error rates on the experimental data

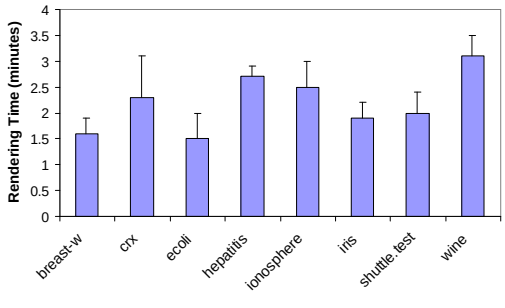

Figure 21: Visual rendering cost (GVR) on the experimental data

First, we use unguided visual rendering (UGV), where no external information is incorporated,

[3] http://disl.cc.gatech.edu/VISTA

to find the visual partition. Unguided visual rendering only depends on the visually observed dense-point areas and the gaps between the areas to discern the clusters and the cluster boundaries. Since there is visual bias on the visualization, the visual rendering sometimes may trap in local minima, where some visual cluster overlaps are not distinguished. It has also been shown that solely depending on visual information to define the clusters is error-prone [30, 59].

We could possibly avoid the local minima by incorporating some external information, either from the result of clustering algorithms or from the domain knowledge. This results in the second rendering method "guided visual rendering (GVR)". In our experiment, ten of labeled items from each cluster are randomly selected serving as the "landmarks" in GVR.

We also compare the visual rendering with two algorithms, K-means and CURE algorithms to see the possible improvement. CURE clustering [53] is recognized as one that can deal with irregular cluster shapes in some degree, and K-means is the most popular algorithm identifying only spherical clusters. Both algorithms use the normalized data as input. The K-means results shown in Figure 20 are the best result in ten runs.

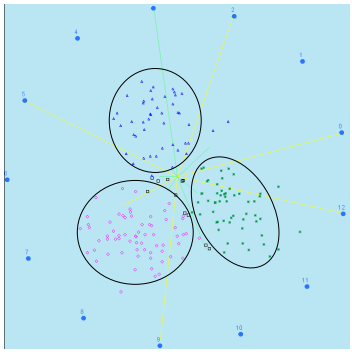

Figure 22: Visualization of "wine" data

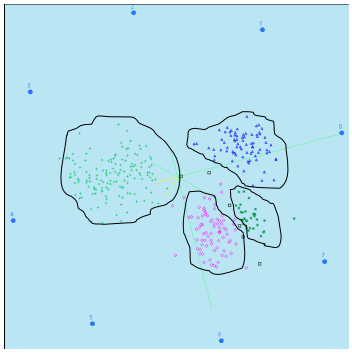

Figure 23: Visualization of "Ecoli" data

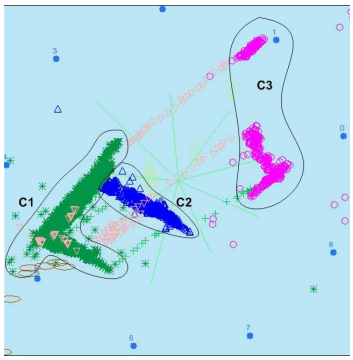

Figure 24: Visualization of "shuttle" data

The experimental result shows that neither CURE or K-means can deal irregularly shaped clusters very well, and UGV may also trap in some local minima for some datasets, for example, "breast-w (breast-cancer-wisconsin)" and "wine" (Figure 22). However, by combining the limited external information we can improve the UGV result to some extent. For example, "Iris" and "Ecoli" (Figure 23) datasets have very clear cluster structure thus the UGV and GVR yield almost the similar results. But in rendering "shuttle" (Figure 24), we need the help of the landmark points to distinguish

the clustering structure, as the domain-specific clusters are formed by merging or splitting some irregular clusters.

In addition, the average interaction time (GVR) of five trained users (Figure 21) indicates that it is not difficult for a trained user to find a satisfactory visualization with the help of the visual rendering rules in section 3.5.2. More detail user studies will be performed to evaluate the visual design.

3.8.3 Outliers and Irregular Cluster Shapes

We have discussed three different labeling algorithms: Representative-Point Based Labeling (RPBL), Centroid-Based Labeling (CBL), and ClusterMap (basic ClusterMap in this set of experiments). In this section we study the performance and error rates in labeling the outliers and irregularly shaped clusters.

We run VISTA to get the ClusterMaps in the map resolution of 688×688. The cost to rebuild a ClusterMap structure in memory is about 340$\sim$360ms, which can be ignored for processing very large datasets. Both DS1 and shuttle datasets show that the estimation of cost is appropriate – all three algorithms are linear complexity and the basic ClusterMap is almost the fastest one (Figure 25).

The DS1 dataset is used to show the effect of outliers on the algorithms. The result shows that the error rate of ClusterMap is much lower than the other two algorithms. By visualizing the labeling result, we observed that CBL can suffer from the variant cluster sizes, the distance between clusters, and the outliers. Particularly, we take a look at the visualization of RPBL result (Figure 26), which clearly shows that the outliers are labeled as the members of the nearby cluster and some points on the cluster boundary are incorrectly labeled too.

"Shuttle" dataset has very irregular clusters. Without the incorporation of domain knowledge, the intermediate clustering will not be satisfactory. If not impossible, it is very difficult for the automated clustering algorithms to incorporate such important external information. We use the shuttle dataset to show that the error in intermediate clustering result might be amplified in labeling phase with the existing labeling algorithms. With the increasing number of labeled records, the error rate of RPBL increases to the level similar to that of CBL, due to its lack of ability dealing with

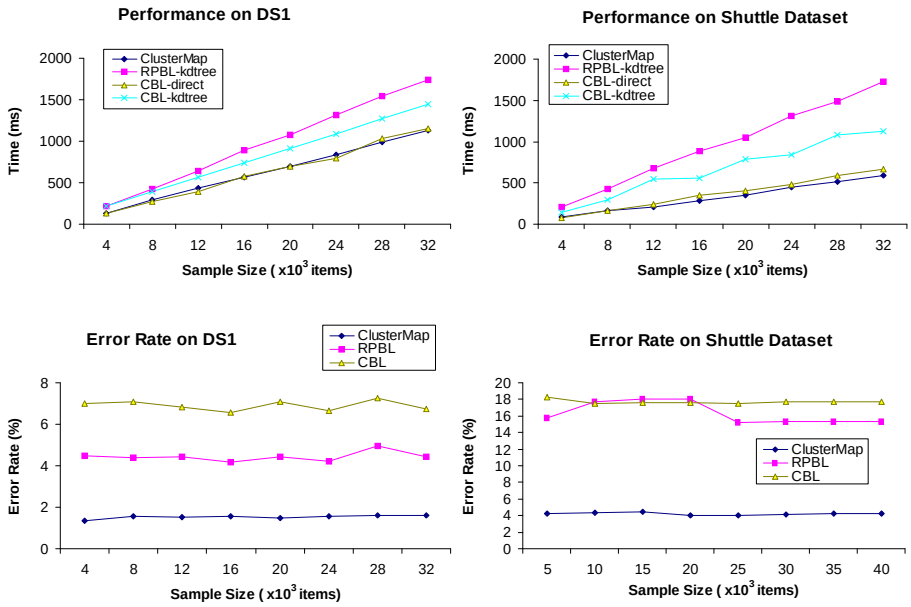

Figure 25: Labeling outliers and irregularly shaped clusters

the very irregular cluster shapes. The basic ClusterMap labeling keeps consistent with the VISTA cluster rendering result and thus has much lower error.

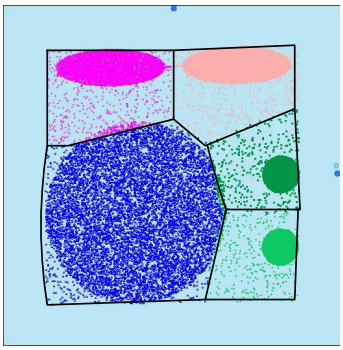

Figure 26: Outliers are labeled as the members of the nearby clusters

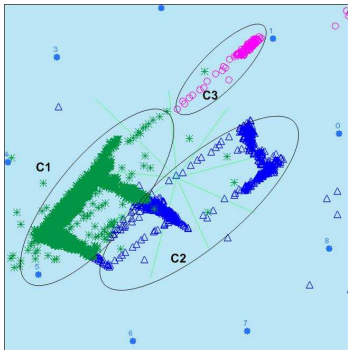

Figure 27: Visualization of the inaccurate RBPL result on Shuttle data

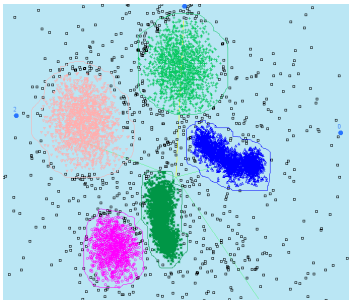

Figure 28: Visualization of 10K samples of LDS

3.8.4 Adaptive ClusterMap on Large Dataset

This experiment on the large dataset LDS mainly shows the scalability and effectiveness of Adaptive ClusterMap algorithm, especially in dealing with outliers and boundary extension.

The progressive plots of time cost and error rate are shown in Figure 29 and 30. The performance curve shows that the two-stage cost of Adaptive ClusterMap labeling will be a little greater than the

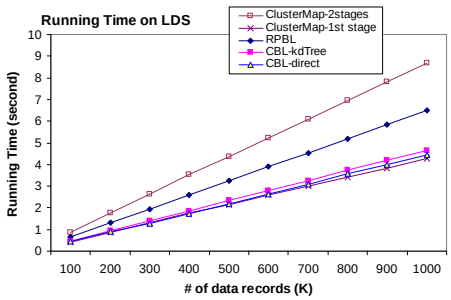

Figure 29: Performance on LDS

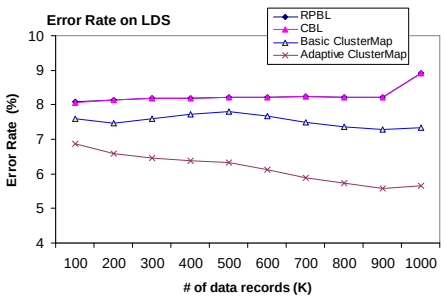

Figure 30: Error rate on LDS

other algorithms, however, it still keeps linear in terms of the number of labeled records. If the second stage is not necessary for some applications as we discussed, the first stage will only cost as little as CBL. Therefore, the Adaptive ClusterMap labeling is still scalable to the number of records.

From the error rate curves (Figure 30), we can observe the difference and the trend between the four algorithms. Basically, RPBL and CBL have higher error than the ClusterMap labeling algorithms due to the lack of ability dealing with outliers and imprecise boundary definition. Since the clusters are well-separated, as the number of labeled records increases, RPBL and CBL basically have the similar labeling results and thus the error rates are very close (they are overlapped in the figure). The basic ClusterMap does not consider the boundary extension, therefore, has higher error than Adaptive ClusterMap. The error of Adaptive ClusterMap tends to decrease with the increasing number of labeled records, because the more the labeled records are incorporated,the better the Adaptive ClusterMap labeling can approximate the real cluster boundary.

this difference. But one of the ClusterMap's advantages is that the users can always visually check and tune the extended ClusterMap after the first stage.

3.9 Exploring Clusters in Very Large Census Dataset: a Comprehensive Example

In this section, we analyze the clusters in a real large dataset – the 1990 Census dataset using the iVIBRATE framework. The procedure revisits most of the concepts, methods, and models we have presented in the previous sections. Concretely, it includes max-min bounds checking and refining in the sampling phase, rendering the sample dataset with the VISTA subsystem, using adaptive ClusterMap labeling to label the entire dataset, and analyzing the small clusters missed by sampling.

Let's start with the description of the dataset.

3.9.1 Dataset

The very large "US Census 1990 Data[4] " is used in this empirical study to show the effectiveness and flexibility of the iVIBRATE framework. This dataset is a discretized version of the raw census data, originally used by the paper [82] in studying the relationship between the sampling approach and the effectiveness of Expectation-Maximization (EM) based clustering algorithms for very large datasets. This dataset is very large in terms of both the number of records and the number of attributes. Although many of the less useful attributes in the raw dataset are dropped, the total number of preserved attributes still reaches 68. It also contains more than 2 million (2,458,284) records, about 352 megabytes in total. Since the dataset is a discretized version, we also run an entropy-based categorical clustering/validation algorithm (ACE algorithm for finding a sub-optimal partition and BkPlot for determining the best Ks) [23] to cross-validate the result in the phase of iterative cluster analysis.

3.9.2 Sampling and Bounds

In section 3.7.1, we have formally analyzed the out-of-bounds rate in terms of the sample size. Note that the analysis is good for any data distributions, which actually results in a quite conservative estimate to the appropriate sample size (around 10,000 sample records) so that the sample bounds are approximately the global bounds. In order to verify this analysis, we try two sets of sample datasets with 5K records and 10K records, respectively. Each set consists of 10 sample datasets, generated by uniformly sampling the entire census dataset. By checking the bounds, we find only two 5K-record sample sets having 1 and 2 attributes, respectively, missing the maximum discretized values, which cause less than 0.1% out-of-bounds rate for the entire dataset. All of the 10K-record datasets include the global bounds. This demonstrates that the estimated threshold in section 3.7.1 is indeed a conservative number. By checking the histograms, we confirm that it is unnecessary to adjust the initial max-min bounds for effective rendering. Therefore, both 5K-record and 10K-record sample datasets should be fine for effective visual rendering.

[4] In UCI KDD Archive http://kdd.ics.uci.edu/databases/census1990/USCensus1990.html

3.9.3 Visual Cluster Rendering

Due to the high dimensionality, manually rendering the data with the dimension-by-dimension method is not recommended for the census dataset. The semi-automated rendering method: automatic random rendering (ARR) followed by automatic dimension-by-dimension rendering (ADDR) (section 3.5.3) is used in rendering a set of 10K-record sample datasets.

Recall that ARR is a randomized process, visual rendering with different rounds of ARR can result in different visualizations. We conduct visual rendering 5 times to see the difference between the rendering results. By comparing the ARR results, the two visually well-separated clusters are confirmed as the same clusters in all of the five renderings. We list some numbers in Table 4 reflecting the consistency between the visualizations. The rendering time is the wall-clock time with the unit of half minute. Due to the high dimensionality and dense point clouds, the average rendering time is about 10 minutes, much higher than those shown in Figure 21. Automatic random rendering (ARR) can quickly identify the sketch of cluster distribution (2 clusters) in about 2 minutes, while automatic dimension-by-dimension rendering (ADDR) should take longer time to refine the visualization. In Table 4, only the number of records in C1 and C2, are shown, the rest of the 10K records are regarded as outliers. ”Shared” represents the shared percentage of the points between the current round of rendering result and the other rounds.

Table 4: Rendering the 10K-record sample census dataset.

No.	ARR	ADDR	C1 (pts)	C1 Shared (%)	C2 (pts)	C2 Shared (%)
1	2.0	8.0	2001	93.0%	7546	87.4%
2	1.5	9.0	2198	84.7%	7610	86.7%
3	2.0	7.0	2070	89.9%	7252	90.9%
4	2.5	8.5	2040	91.2%	7403	89.1%
5	1.5	7.0	2189	85.0%	7508	87.8%

We show three of the five visualizations in Figure 31, 32, and 33. All three results clearly show the two major clusters. Figure 31 and 32 also show that C2 could potentially have two subclusters. Cluster validation with the entropy criterion [23] shows that the “best K” for clustering census dataset should be 3 or 2 (Figure 36), which implies that the initial visual clue about the possible existence of three clusters could be true. Similarly, Figure 32 and 33 show that C1 may have some fine structure, which we can also further refine if necessary.

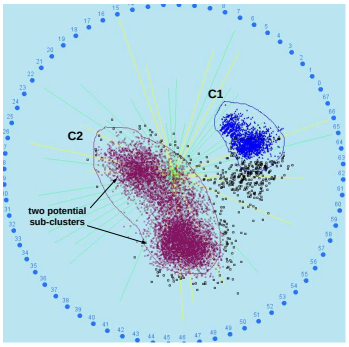

Figure 31: Result of visual rendering 1

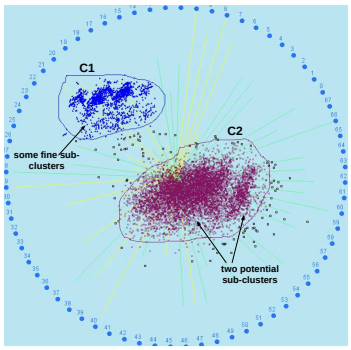

Figure 32: Result of visual rendering 2

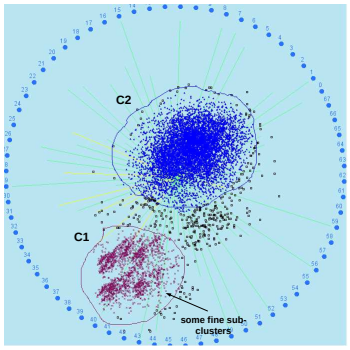

Figure 33: Result of visual rendering 3

By exploring the clusters C1 and C2 separately, we identify that C2 indeed contains two sub-clusters. We do not show the separated rendering result but the refined visualization of the three clusters only (Figure 34). The result is confirmed by the entropy-based cluster validation method. Figure 35 shows that the visually separated clusters match well with the clusters labeled by the entropy-based ACE categorical clustering algorithm [23]. Note that from the algorithmic result, we are not able to identify the outliers and the initial cluster boundary, but with the VISTA system we are able to interactively define the initial cluster boundary. Figure 34 also shows the initial cluster boundary which is used to generate the ClusterMap for the labeling phase.

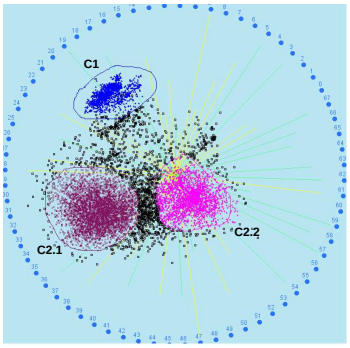

Figure 34: Final result of visual rendering with initial boundary – three clusters

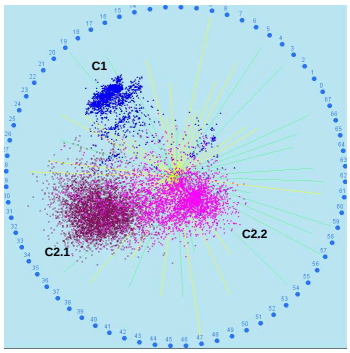

Figure 35: Visualization of ACE clustering result

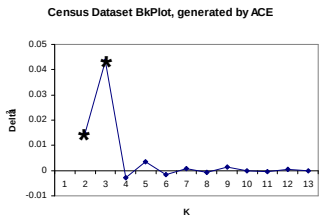

Figure 36: The "Best K" plot for census dataset

In summary, in the above procedure, we show that the combination of the automatic clustering/validation algorithms and the visual rendering method can really help to cross-validate and improve each other. Therefore, cluster analysis under the iVIBRATE framework can provide truly

insightful results with much higher confidence level.

3.9.4 ClusterMap Labeling

Visually clustering the census data under the iVIBRATE framework is a procedure of visual data exploration (Unguided Visual Rendering as defined earlier), in the sense that no initial clustering clues, such as domain knowledge, are provided. For LDS dataset in Section 3.8.4, we can predefine exact cluster labels with little error and compare the labeling results with the exact cluster labels. However, because of the lack of control labels for the census dataset, we need to change the way of evaluating the labeling algorithms.

As we have initially observed, Adaptive ClusterMap labeling, together with the monitoring tool "ClusterMap Observer", gives much less labeling error and the result becomes very closer to the exact cluster labels with the increase of labeled records. Without the exact control labels for the census dataset, we want to see the difference between the results of other labeling algorithms and the Adaptive ClusterMap only, which should also be consistent with our interpretation in Section 3.8.4.

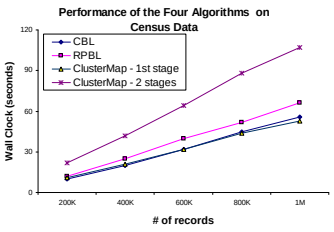

Figure 37: Cost of labeling the census dataset

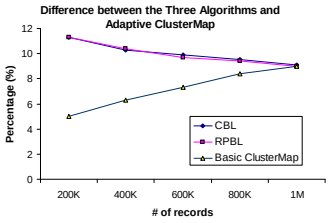

Figure 38: Difference of the labeling results

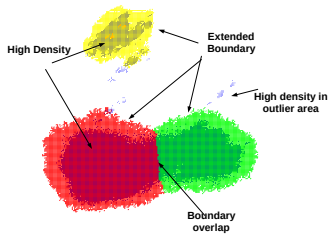

Figure 39: The ClusterMap after labeling 1M records

Figure 37 shows the performance curve of the three labeling algorithms similar to what we observed for the LDS dataset. Figure 38 shows the percentage of difference between the compared algorithms and the adaptive ClusterMap labeling. Since the three clusters are almost regularly shaped and distributed, RPBL and CBL have no big difference as shown in Figure 38. The difference between RPBL/CBL and Adaptive ClusterMap tends to decrease with the increase number of labeled items, due to the automatic boundary extension including more and more points previously regarded

as outliers. At the stage a large number of records, for instance, 1 Million records, are labeled, most of the difference should come from outliers. Similarly, the increase of difference between the basic ClusterMap and the adaptive version is solely caused by boundary extension.

Figure 39 demonstrates the cluster and boundary definition after labeling 1 million records. The boundary is evenly extended around the original boundary, therefore, the initial ClusterMap definition is very good. C2 and C3 tend to merge, but C1 is still clearly separated from C2 and C3. Whether to merge C2 and C3 or not may depend on the domain knowledge.

3.9.5 Exploring the Small Clusters

Some outliers emerge with high density as labeled in Figure 39. Therefore, it might be worthwhile to visualize the outliers separately. Labeling the first million records results in 70632 labeled outliers, about 7% of the total. We sample the outliers again to gain a 10K-record sample set, which is visualized as Figure 40. The initial visualization with the same parameters used in labeling clearly shows that clusters could emerge in the areas A1, A2, A3. By rendering this dataset, we find 5 clusters (the left panel in Figure 40). O1 is mapped to the boundary of the original cluster C2, therefore it should be outliers around C2. O2 and O3 are mapped to the areas A2 and A3 respectively, showing the evidence of new clusters in these areas. O4 and O5 are mapped to the same area A1, which visualizes the delicate structure hiding in A1.

Observing the proximity of the small clusters to the three identified main clusters in Figure 40, we find that O4 and O5 might have close connection with the main cluster C1, and O2 and O3 seem some kind of extension of the main cluster C2.2. Merging them to the main clusters or not should depend on the domain knowledge.

In summary, the adaptive ClusterMap labeling and the small cluster analysis with the census dataset demonstrate the effectiveness and flexibility of iVIBRATE framework in processing very large datasets. With the iVIBRATE framework, irregularly shaped clusters, domain-specific clustering structure, cluster-boundary extension, outlier labeling, and small cluster detection, can be performed and monitored with the help of visualization, which greatly improves the precision of clustering result and increases the confidence about the result.

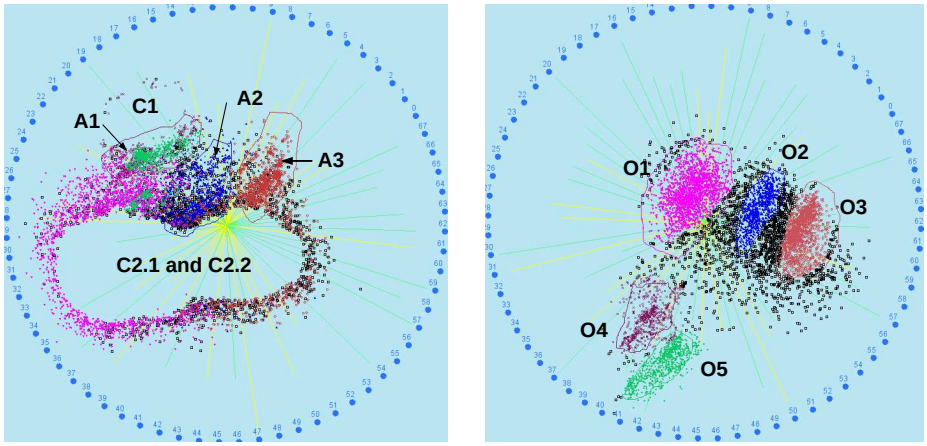

Figure 40: Exploring possible small clusters hiding in outliers

CHAPTER IV

"BEST K": THE CRITICAL CLUSTERING STRUCTURE IN CATEGORICAL DATA

In last chapter, we have discussed the challenges in clustering large *numerical* datasets. An interactive visualization model, VISTA, is proposed to utilize the density features of clusters in Euclidean space. VISTA model is a dimensional-parameter-tunable projective mapping model, especially good for finding and defining irregularly shaped clusters and domain-specific clustering structures. Different from numerical data, which can be easily embedded into Euclidean space, categorical data does not have intuitive distance meaning and there is no well-agreed distance measure for categorical data. Since categorical datasets widely exist in real-world applications, it is necessary to develop the clustering methods and address the related challenges for categorical datasets.

Among the clustering related issues, determining the best number of clusters is a very challenging problem. For numerical data clustering, several statistical methods have already been developed to address this problem, such as intra/inter-cluster similarity based measures [56], and the traditional model-based AIC/BIC methods [57]. In particular, VISTA visual cluster rendering system can also be used to visually determine the best K number of clusters, and it has been proved better than the traditional methods for irregularly shaped clusters [22]. However, VISTA can not be applied to categorical data since Euclidean distance is not suitable for categorical data. In this chapter, we develop a new method for determining the best K for categorical clustering. Although this method is more related to information theoretic concepts, we also utilizes some properties of curves (see Section 2.1) to define the Best K Plot (see Section 4.5).

This chapter is organized as follows. Section 4.1 introduces the challenges in categorical clustering and cluster validation. The scope and contributions of this work in given in Section 4.2. Section 4.3 reviews some related work in categorical clustering and cluster validation. Section 4.4 sets down the basic concepts and notations. Section 4.5 introduces the BKPlot method for determining the best

Ks and the intuition behind the method. Section 4.6 uses the concept of "incremental entropy" to develop the hierarchical clustering algorithm ACE for generating high-quality approximate BKPlots. In section 4.7 and 4.8, we analyze the sample approximate BKPlots for large datasets, and propose a testing method for determining the existence of significant clustering structure. The experimental results are presented in section 4.9.

4.1 Clustering and Cluster Validation for Categorical Data

Clustering techniques for categorical data are very different from those for numerical data in terms of the definition of similarity measure. Traditionally, categorical data clustering is merged into numerical clustering through a data preprocessing stage [66]. In the preprocessing, numerical features are extracted/constructed from the categorical data, or a conceptual similarity function between data records is defined based on the domain knowledge.

However, meaningful numerical features or conceptual similarity are usually difficult to extract at the early stage of data analysis, because we have little knowledge about the data. It has been widely recognized that directly clustering the raw categorical data is important for many applications. Examples include environmental data analysis [107], market basket data analysis [1], DNA or protein sequence analysis [14], and security [11]. Therefore, recently there are increasing interests in clustering categorical data [62, 54, 48, 49, 12, 35, 7, 78].

Cluster Validation for Categorical Data Different clustering algorithms usually do not generate the same clustering result for the same dataset, and we need cluster validation methods to evaluate the quality of clustering results [93, 65, 56]. Formally, there are three main issues in cluster validation: 1) how to evaluate the quality of different partition schemes generated by different clustering algorithms for the same dataset, with a fixed K number of clusters; 2) how to determine whether there is significant clustering structure in the dataset; 3) how to determine the best number of clusters (the "best K"), if there is inherent significant clustering structure in the dataset.

Due to the lack of the distance meaning between categorical values, the techniques used in cluster validation for numerical data are not applicable to categorical data. Without reasonable numerical feature extraction/construction for a given categorical dataset, the general distance functions are usually inapplicable. As a result, no geometry/density-based validation method is appropriate

in validating the clustering result for categorical data. Surprisingly, to our knowledge, there is no literature satisfactorily addressing the cluster validation problems for categorical data.

Entropy for Categorical Clustering One way to address the similarity problem for categorical data is to use set-based similarity measures, for example, the *entropy* [31] based measures. Originated from information theory, entropy has been applied in both pattern discovery [17] and numerical clustering [27]. Recently, there have been some efforts in applying entropy and the related concepts in information theory to clustering categorical data [12, 78, 35, 7]. Initial results have shown that entropy criterion can be very effective in clustering categorical data.

Entropy-based similarity measures evaluate the orderliness of a given cluster. In entropy-based categorical clustering, the quality of clustering result is naturally evaluated by the entropy of all clusters [12, 78], namely, the *expected entropy*. The lower the expected entropy is, the more ordered the clusters are. While it is intuitive to evaluate the overall orderliness of a clustering result with expected entropy, can entropy also be used to identify the best K number of clusters? And, can it be used to determine whether there is significant clustering structure in a given dataset?

4.2 The Scope and Contributions of This Chapter

We try to answer the above cluster validation problems based on the entropy difference between the optimal clustering structures. Intuitively, if the best clustering structure has K clusters, fitting the data from K clusters into $K-1$ clusters will seriously disturb the clustering structure, while the change of optimal $K+1$ clusters to K clusters should be much less distinctive. This leads us to explore the property of *optimal neighboring clustering structures* with K and $K+1$ clusters, respectively, K varying from one to a small number, e.g., $K < 20$, for the primary clustering structures. This optimality is evaluated with expected entropy. Briefly, we identify and interpret that the *similarity between the optimal clustering structures* is the key to find the critical clustering structures. The "*Best-K Plot* (BKPlot)" method is proposed to conveniently capture the dramatic difference between the optimal clustering structures.

However, optimal BKPlots are based on the optimal clustering results, which are intractable due to the NP-hard complexity of entropy minimization. Generating high-quality approximate BKPlots becomes a significant problem in practice. We address this problem with two aspects. First, we

propose a new inter-cluster similarity measure *Incremental Entropy (IE)*. Based on IE, the Agglomerative Categorical clustering algorithm with Entropy criterion ("ACE for short) is developed for generating reliable approximate BKPlots. The initial experimental results show that ACE algorithm can generate high-quality approximate BKPlots, compared to other entropy-criterion based algorithms. Second, for large datasets, we develop the theory of *sample approximate BKPlot*, which describes how to consistently identify the Best Ks based on small sample datasets with the ACE algorithm.

There are also datasets having no significant clustering structure, such as uniformly distributed data. By exploring the characteristics of the BKPlots for the typical no-cluster datasets, we provide a testing method for determining whether a given dataset has significant clustering structure.

In summary, we propose a framework for determining the critical clustering structure in categorical datasets, which consists of four main components

1. the basic BKPlot method for determining the best Ks;
2. the entropy-based inter-cluster similarity measure and the algorithm for generating reliable approximate BKPlots;
3. the theory of sample approximate BKPlot for large datasets;
4. a testing method for determining whether there is significant clustering structure for a given dataset.

4.3 Related Work

While many numerical clustering algorithms [65, 66] have been published, only a handful of categorical clustering algorithms appear in literature. The general statistical analysis of categorical data was introduced in [6]. Although it is unnatural to define a distance function between categorical data records or to use the statistical center (the mean) of a group of categorical items, there are some algorithms, for example, K-Modes [62] algorithm and ROCK [54] algorithm, trying to fit the traditional clustering methods into categorical data. However, since the numerical similarity/distance function may not describe the categorical properties properly and intuitively, it leaves little confidence to the clustering result.

CACTUS [48] adopts the linkage idea from ROCK and names it “strong connection”. However, the similarity is calculated by the “support”. A cluster is defined as a region of attributes that are pairwise strongly connected.Similarly, the concept of “support” or linkage is still indirect in defining the similarity of categorical data, and unnecessarily makes the clustering process complicated.

Gibson et al. introduced STIRR [49], an iterative algorithm based on non-linear dynamical systems. STIRR represents each attribute value as a weighted vertex in a graph. Starting with the initial conditions, the system is iterated until a “fixed point” is reached. When the fixed point is reached, the weights in one or more of the “basins” isolate two groups of attribute values on each attribute. Even though it is shown that this algorithm works for the experimental datasets having two partitions, it is challenging to determine the optimal number of clusters solely with this algorithm.

Cheng et al. [27] applied the entropy concept in numerical subspace clustering, and Coolcat [12] introduced the entropy concept into categorical clustering. Coolcat is kind of similar to KModes. However, Coolcat assigns the item to a cluster that minimizes the expected entropy. Considering the cluster centers may shift, a number of worst-fitted points will be re-clustered after a batch. Even though Coolcat approach introduces the entropy concept into its categorical clustering algorithm, it did not consider the problem of finding the optimal number of categorical clusters. Some closely related work also borrows concepts from information theory, including Co-clustering [35], Information Bottleneck [101], LIMBO [7], and Cross-association [20]. All of the above approaches are closely related if considered under the probabilistic clustering framework [78].

Most of the recent research in categorical clustering is focused on clustering algorithms. Surprisingly, there is little research concerning about the cluster validation problems for categorical datasets. The “Best K” problem has been discussed in terms of numerical data, especially with the mixture models. AIC and BIC [57] are the major model-based criteria for determining the best mixture model (with the “Best K”). They have been used widely in validating the Gaussian mixture based numerical clustering. They are supposed to also be effective if appropriate mixture models are used for categorical data. Multinomial mixture is usually used to model categorical data. As model-based clustering fits data into assumed models, exceptions can happen when the assumed model is not appropriate. We compare the Multinomial-mixture based BIC criterion and our BKPlot method in experiments. Results show that the BKPlot method has unique advantages in determining the

best Ks: 1) it can determine all the best Ks for multi-layer clustering structures, while BIC cannot; 2) cluster overlapping can create some difficulty for BIC, but BKPlot can successfully handle it.

4.4 Basic Notations and Concepts

We define the notations particularly used in this chapter and then introduce the entropy-based clustering criterion. Some basic properties about the entropy criterion will be presented in the later sections.

4.4.1 Basic Entropy Definition

Consider that a dataset $\mathbb{S}$ with N records and d columns, is a sample set of the discrete random vector $X = (x_1, x_2, \ldots, x_d)$. For each component x_j, $1 \leqslant j \leqslant d$, x_j takes a value from the domain A_j. A_j is conceptually different from $A_k (k \neq j)$. There are a finite number of distinct categorical values in $domain(A_j)$ and we denote the number of distinct values as $|A_j|$. Let $p(x_j = v)$, $v \in A_j$, represent the probability of $x_j = v$, we have the classical entropy definition [31] for a dataset.

$$H(X) = \sum_{j=1}^{d} H(x_j) = -\sum_{j=1}^{d} \sum_{v \in A_j} p(x_j = v) \log_2 p(x_j = v)$$

Since $H(X)$ is estimated with the sample set $\mathbb{S}$ in practice, we define the estimated entropy as $\hat{H}(X) = H(X|\mathbb{S})$.

$$\hat{H}(X) = H(X|\mathbb{S}) = -\sum_{j=1}^{d} \sum_{v \in A_j} p(x_j = v|\mathbb{S}) \log_2 p(x_j = v|\mathbb{S})$$

we also define the **column entropy** of A_j as

$$H(A_j|\mathbb{S}) = -\sum_{v \in A_j} p(v|\mathbb{S}) \log_2 p(v|\mathbb{S})$$

In practice, we use the following adjusted expected entropy to avoid the domination of skewed high column cardinalities [86].

$$\hat{H}(X) \quad = \quad \sum_{j=1}^{d} \frac{1}{d \log_2 |A_j|} H(A_j|\mathbb{S})$$

Suppose the dataset $\mathbb{S}$ is partitioned into K clusters. Let $C^K = \{C_1, \ldots, C_K\}$ represent a **partition**, where C_k is a cluster and n_k represent the number of records in C_k. Thus, the **cluster entropy** of C_k is the dataset entropy $\hat{H}(C_k)$.

The classical entropy-based clustering criterion tries to find the optimal partition, C^K, which maximizes the following entropy criterion [15, 19, 78].

$$O(C^K) = \frac{1}{d}\left(\hat{H}(X) - \frac{1}{n}\sum_{k=1}^{K} n_k \hat{H}(C_k)\right)$$

Since $\hat{H}(X)$ is fixed for a given dataset $\mathbb{S}$, maximizing $O(C^K)$ is equivalent to minimizing the item $\frac{1}{n}\sum_{k=1}^{K} n_k \hat{H}(C_k)$, which is named as the **expected entropy** of partition C^K. Let us denote it as $\bar{H}(C^K)$. For convenience, we also name $n_k \hat{H}(C_k)$ as the **weighted entropy** of cluster C_k.

Li et al [78] showed that the minimization of expected-entropy can be unified into probabilistic clustering framework, and closely related to many important concepts in information theory, clustering and classification, such as Kullback-Leibler Measure [31], Maximum Likelihood [76], Minimum Description Length [31], and dissimilarity coefficients [13]. Entropy criterion is especially good for categorical clustering due to the lack of intuitive definition of distance for categorical values. While entropy criterion can also be applied to numerical data [27], it is not the best choice for capturing all of the geometric properties a numerical dataset may have.

4.4.2 Incremental Entropy

Individually, cluster entropy cannot determine the structural difference between clusters. However, we observe that the structural difference can be observed by mixing (merging) two clusters. By entropy definition, the structural characteristic of a dataset is determined by the value frequencies, i.e., $p(x_j = v|C_k)$, in each column. Intuitively, mixing two clusters that are similar in the inherent structure will not change the value frequencies, thus, not change the expected-entropy of the partition as well. However, merging dissimilar ones will inevitably change the value frequencies, increasing the expected-entropy. Therefore, the increase of expected entropy in merging clusters has some correlation with the similarity between clusters.

By the definition of expected-entropy, after merging two clusters in a partition the difference of expected-entropy can be equivalently evaluated by the difference between the weighted entropies, i.e., $(n_p + n_q)\hat{H}(C_p \cup C_q)$ and $n_p\hat{H}(C_p) + n_q\hat{H}(C_q)$. We have the following first result about weighted entropies.

Proposition 4. $(n_p + n_q)\hat{H}(C_p \cup C_q) \geqslant n_p\hat{H}(C_p) + n_q\hat{H}(C_q)$

SKETCH PROOF. This proposition formally states that mixing two clusters will not reduce the entropy. The first step of the proof is to expand both sides of the formula with the entropy definition.

$$
\begin{aligned}
- & \sum_{j=1}^{d} \frac{1}{d \log_2 |A_j|} \sum_{v \in A_j} (n_p + n_q) p(x_j = v | C_p \cup C_q) \cdot \log_2 p(x_j = v | C_p \cup C_q) \geqslant \\
- & \sum_{j=1}^{d} \frac{1}{d \log_2 |A_j|} \sum_{v \in A_j} n_p p(x_j = v | C_p) \log_2 p(x_j = v | C_p) \\
- & \sum_{j=1}^{d} \frac{1}{d \log_2 |A_j|} \sum_{v \in A_j} n_q p(x_j = v | C_q) \log_2 p(x_j = v | C_q)
\end{aligned} \tag{13}
$$

It is straightforward to prove that the above formula is true if the following relation is satisfied for each value v in each column A_j. Namely, if we can prove that for each categorical value in each column the following formula is true, then the proposition is established.

$$
\begin{aligned}
& n_p p(x_j = v | C_p) \log_2 p(x_j = v | C_p) + n_q p(x_j = v | C_q) \log_2 p(x_j = v | C_q) \\
& \qquad \geqslant (n_p + n_q) p(x_j = v | C_p \cup C_q) \cdot \log_2 p(x_j = v | C_p \cup C_q)
\end{aligned} \tag{14}
$$

Without loss of generality, suppose C_p having x rows and C_q having y rows with value v at j-th attribute, $x, y > 0$ (if $x = 0$ or $y = 0$, the equation is trivially satisfied), i.e., $p(x_j = v|C_p) = \frac{x}{n_p}$ and $p(x_j = v|C_q) = \frac{x}{n_q}$. Then, the formula 14 can be transformed to $(\frac{x}{n_p})^x \cdot (\frac{y}{n_q})^y \geqslant (\frac{x+y}{n_p+n_q})^{(x+y)}$. Since x, y, n_p, n_q are positive integers, let $x = s \cdot y$ and $n_p = r \cdot n_q, (s, r > 0)$, then the formula (14) is further transformed to $\frac{r^s}{(1+r)^{s+1}} \leqslant \frac{s^s}{(1+s)^{1+s}}$. Given that $\frac{s^s}{(1+s)^{1+s}}$ is the maximum value of the function $f(r) = \frac{r^s}{(1+r)^{s+1}} (r, s > 0)$ (at $r = s$), thus the formula (14) is established. We can now conclude that the formula (13) is true and so is Proposition 4. □

We define the key concept "*Incremental Entropy* (IE)" based on the above proposition.

Definition 11. *Incremental Entropy reflects the structural difference between two clusters, which is quantified by* $IE(C_p, C_q) = (n_p + n_q)\hat{H}(C_p \cup C_q) - (n_p \hat{H}(C_p) + n_q \hat{H}(C_q))$.

From the proof of Proposition 4, it follows that if the two clusters have the *identical structure* – for every categorical value v_i in every attribute x_j, $1 \leqslant i \leqslant |A_j|$, $1 \leqslant j \leqslant d$, $p(x_j = v_i|C_p) = p(x_j = v_i|C_q)$ is satisfied, $IE(C_p, C_q) = 0$. Interestingly, identical structure does not concern the

size of the clusters. Thus, it intuitively implies that sampling will be effective when IE is used as a clustering criterion.

Incremental entropy is a critical measure in our method. We will use this similarity measure to construct a hierarchical clustering algorithm in order to generate reliable results for determining the best Ks. The relationship between cluster merging and similarity can also help us understand the method to find the best Ks in next section.

For reference convenience we summarize some important notations in Table 5.

Table 5: Notations.

notation	description
d, N, n	d: the number of attributes, N: the size of dataset, n: the sample size
A_j, $\|A_j\|$	A_j represents attribute j, and $\|A_j\|$ is the number of distinct categorical values in this attribute
$\hat{H}(C_k)$	entropy of cluster C_k
$\hat{H}(C_{n,k})$	entropy of cluster C_k with sample size n
$\hat{H}(A_j\|C_k)$	the column entropy of column A_j in cluster C_k
$\bar{H}(C^K)$	expected entropy of a K-cluster partition
$\bar{H}_{opt}(C^K)$	the minimum expected entropy of all K-cluster partitions
$I(K)$	the entropy difference between a pair of optimal neighboring partitions, i.e., $\bar{H}(C^K) - \bar{H}(C^{K+1})$
$I(n, K)$	the approximate I(K) with sample size n
$IE(C_p, C_q)$	incremental entropy between clusters C_p and C_q
$B(K)$	the BKPlot function, derived from I(K)
$B(n, K)$	the approximate B(K) with sample size n

4.5 Finding the Best K with Entropy Criterion

Traditionally, the Best Ks for numerical data clustering are identified with statistical index curves, based on geometry and density properties of the dataset [93, 56] or likelihood [57]. Depending on the different property of the index curve, the Ks at peaks, valleys, or distinguished "knees" on the curve, may be regarded as the candidates of the optimal number of clusters (the best Ks). *Are there entropy-based such curves indicating the significant clustering structures for categorical data?*

The first thought might be investigating the curve of expected entropy for the optimal partitions. We define

Definition 12. *An optimal partition of K clusters is a partition that results in the minimum expected entropy among all K-cluster partitions.*

The expected entropy of the optimal partition is denoted by $\bar{H}_{opt}(C^K)$. Our result shows that the $\bar{H}_{opt}(C^K)$ curve is often a smoothly decreasing curve without distinguished peaks, valley, or

knees (Figure 41). However, we can actually obtain some information between the neighboring optimal partitions (with K and $K+1$ clusters respectively) with the concept of entropy difference. Concretely, the difference of neighboring expected-entropies (Figure 42) can be used to indicate the critical clustering structures. This relationship can be intuitively illustrated and understood by merging similar clusters in an optimal partition. The entropy-difference curve often shows that the similar partitions with different K are at the same "plateau". From plateau to plateau there are the critical points implying the significant change of clustering structure, which can be the candidates of the best Ks. Before going to details, we first give some entropy properties of optimal partitions.

4.5.1 Entropy Properties of Optimal Partitions

Given the number of clusters, K, there is at least one optimal partition with minimum expected entropy $\bar{H}_{opt}(C^K)$. There are several properties about $\bar{H}_{opt}(C^K)$.

First of all, $\bar{H}_{opt}(C^K)$ is bounded. It is easy to see that $\bar{H}(C^K)$ is less than the dataset entropy $\hat{H}(X)$. $\bar{H}(C^K)$ is maximized when $K = 1$ – all data points are in the same cluster. We also have $\bar{H}(C^K) \geqslant 0$ as the entropy definition implies. The zero entropy $\bar{H}(C^k)$ is reached at $k = N$, when each record is a cluster. Therefore, $\bar{H}_{opt}(C^K)$ is bounded by $[0, \hat{H}(X)]$.

We can also derive the relationship between the optimal partitions with any different number of clusters, K and L, $K < L$, with the help of Proposition 4.

Proposition 5. $\bar{H}_{opt}(C^K) \geqslant \bar{H}_{opt}(C^L)$, *when* $K < L$

SKETCH PROOF. Let a L-cluster partition C_0^L be formed by splitting the clusters in the optimal K-cluster partition. With Proposition 4 and the definition of optimal partition, we have

$$\bar{H}_{opt}(C^K) \geqslant \bar{H}(C_0^L) \geqslant \bar{H}_{opt}(C^L)$$

□

Proposition 5 shows that the optimal expected-entropy decreases with the increasing of K, which meets the intuition well. It is hard to describe the curve with a function of closed form in terms of K. As our experimental result shows, it is often a negative logarithm-like curve (Figure 41). This curve implies that, 1) it is highly possible that the best K is not unique in terms of entropy criterion, and 2) with only expected-entropy curve, we can not clearly identify the significant

clustering structures.

4.5.2 Understanding the Similarity of Neighboring Partitions

In this section, we focus on the similarity between the neighboring optimal partitions. We formally define this similarity with the entropy difference between the neighboring partitions as

$$I(K) = \bar{H}_{opt}(C^K) - \bar{H}_{opt}(C^{K+1})$$

There are two aspects to capture this similarity. One aspect is the absolute value of $I(K)$, which indicates how much the clustering structure is changed. The other aspect is the difference between $I(K)$ and $I(K+1)$, which indicates whether the consecutive changes to the clustering structure are similar. Since it is not easy to understand the change between the optimal partitions, we use a cluster merging scheme, which will be described in Section 4.6, to demonstrate the two aspects of similarity.

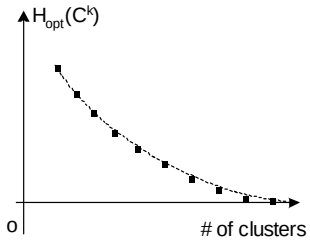

Figure 41: Sketch of expected entropy curve.

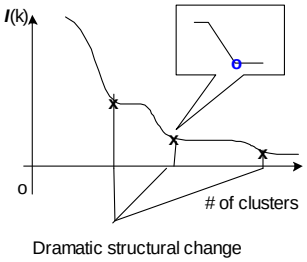

Figure 42: Sketch of entropy-difference curve of neighboring partitions.

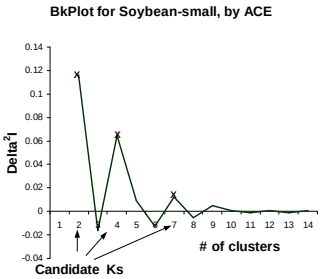

Figure 43: Finding the best k with BKPlot (example of soybean-small data).

- First, small $I(K)$ means high similarity between the neighboring partitions. We can understand this by merging the similar clusters in K+1-cluster partition to form K-cluster partition. Merging identical clusters introduces zero increase of entropy and the clustering structure is not changed at all. Similarly, small increasing rate between two neighboring schemes implies that the reduction of number of clusters does not introduce significant change to the clustering structure.

- For large change of expected entropy, we first consider the meaning of large $I(K)$. If the expected-entropy increases a lot from K+1 to K, this reduction of number of clusters should

introduce considerable impurity into the clusters and thus the clustering structure can be changed significantly. However, whether this change is globally distinguishable from others depends on the further comparison between the continuous changes. Consider $I(K)$ as the amount of extra impurity introduced from $K+1$-cluster partition to K-cluster partition. If $I(K) \approx I(K+1)$, i.e. K-cluster partition introduces similar amount of impurity as K+1-cluster scheme does, we consider that the clustering structure is *similarly* changed from K+1-cluster scheme to K-cluster scheme.

An example of "similar merge" in Figure 44 can well demonstrate similar changes vs. significant changes of clustering structure. We use icons to conceptually represent categorical clusters. The shape and the size of an icon represent the structure and the size of the cluster, respectively. Suppose that the initial state with $C1 - C5$ are the optimal partition of K+2 clusters, among which C1, C2, and C3 are in a similar clustering structure as shown in Figure 44. Now, we try to form sub-optimal partitions by merging the similar clusters. As Proposition 4 shows, merging the clusters with similar clustering structure will result in small IE, which means small entropy difference between the two neighboring partitions. Suppose that, $\hat{H}(C^{K+1})$ is approximately minimized by merging C1 and C2, and $\hat{H}(C^{K})$ by merging C1+C2 and C3. Since the three clusters are in a similar structure, the consecutive two merge operations result in similar $I(K)$ and $I(K+1)$. The resultant clustering structures should not be distinguishable from each other. On the other hand, reducing the number of clusters further from K to $K-1$ will change the clustering structure a lot and inevitably bring more impurity into the partitions. As a result, $I(K-1)$ will be much larger than $I(K)$ and $I(K+1)$. We can see that K becomes a significant point where a series of similar clustering structures are changed to a significantly different clustering structure. Therefore, the "knees" on the $I(K)$ curve can provide substantial information about similar changes or significant changes of clustering structure.

In practice, if a dataset has significant clustering structure, we can find a series of neighboring "stable" schemes, which result in similar $I(K)$, and we may also find the critical points where a series of "stable" schemes become "less stable" when K is reduced (Figure 42). All of the critical points should be the candidates of the best Ks and could be interesting to cluster analysis.

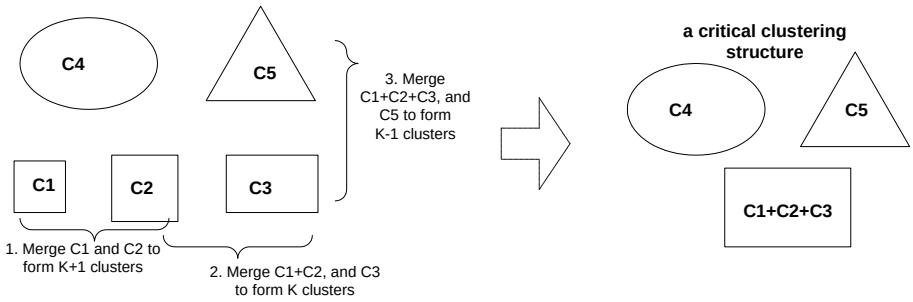

Figure 44: Similar merges with $I(K) \approx I(K+1)$, but $I(K-1) \gg I(K)$

The common way to mathematically identify such critical knees on the $I(K)$ curve is to find the peaks/valleys of the second-order difference of the curve. Specifically, since $I(K)$ curve consists of a set of discrete points, we define the second-order difference of the curve as

$$B(K) = \delta^2 I(K) = \delta I(K-1) - \delta I(K)$$

and $\delta I(K) = I(K) - I(K+1)$ to make K aligned with the critical points. For convenience, we name the $B(K)$ curve as the "Best-k Plot (BKPlot)" (Figure 43). Notice that dramatic structure change happens only at $I(K) > I(K+1)$. $I(K) < I(K+1)$ means the structure change is slowing down when the number of clusters is reduced from $K+1$ to K, and we need to keep looking at the range $2 \leq k < K$ to find more dramatic changes. Therefore, we need only to look at the peaks of BKPlot.

The basic idea of BKPlot method is based on the optimal clustering results (the minimum expected entropy for the k-cluster partition). Thus, it is challenging to generate the optimal BKPlot since obtaining the optimal clustering result is computationally intractable for even a small dataset. However, it is practical and feasible to find approximate BKPlots. It is important to note that finding approximate BKPLots is different from finding the approximately optimal clustering results, even though they are tightly related. High quality BKPlots are the approximate BKPlots that can consistently and correctly identify the candidate best Ks. Although there are many clustering algorithms proposed so far, none have designed with the above objectives in mind. Hence, we develop a method for generating high-quality approximate BKPlots. We call it Agglomerative Categorical clustering algorithm with Entropy criterion and ACE for short.

4.6 Generating High-quality Approximate BKPlot: ACE Algorithm

In this section, we utilize the proposed similarity measure – Incremental Entropy to construct a hierarchical clustering algorithm (ACE) for generating reliable approximate BKPlots. We first briefly introduce the optimization steps in ACE algorithm, and then analyze the property of sample BKPlots generated by ACE algorithm in next section. The quality of BKPlots generated by ACE will be evaluated through experiments.

ACE algorithm follows the framework of standard hierarchical clustering algorithm with two unique features: (i) the incremental entropy is used as the inter-cluster similarity measure; and (ii) the optimization steps in calculating the incremental entropy based similarity are designed specifically for generating reliable high-quality BKPlots. Experimental results show that ACE algorithm is the most effective algorithm for generating high-quality approximate BKPlots, compared to other existing algorithms optimizing the same expected entropy criterion, such as Monte-Carlo [78] and Coolcat [12].

Furthermore, ACE algorithm has a nice property in generating sample BKPlots for handling large datasets. The mean of sample BKPlots generated by ACE algorithm on sample datasets is an unbiased estimator of the original BKPlot generated by ACE on the entire dataset. We will prove this property in next section.

4.6.1 ACE Algorithm

While the traditional hierarchical algorithms for numerical clustering need to explicitly define the inter-cluster similarity with "single-link", "multi-link" or "complete-link" methods [65], incremental entropy is a natural cluster-based similarity measure, ready for constructing a hierarchical clustering algorithm. ACE algorithm can be briefly described in the following steps.

1. It begins with the extreme scenario where each record is a cluster, and follows by an iterative process to perform cluster merges.

2. The algorithm finds a pair of clusters C_p and C_q that are the most similar, i.e. the incremental entropy $IE(C_p, C_q)$ is minimum among all pairs of clusters.

3. Merge C_p and C_q. Update the data structures that keep track of the closest cluster for each

given cluster. Return to step 2 until K=2.

We use $IE^{(K)}$ to denote the IE value in forming K-cluster partition from K+1-cluster partition. Maintaining the minimum IE for each step is the most costly part of this algorithm. We briefly describe the data structures and the optimization steps applied in this algorithm.

ACE uses three structures to maintain the minimum IE value in step 3. *summary table* for convenient counting of occurrences of values, since IE calculation involves primarily the weighted entropy calculation. IE*-table* for bookkeeping $IE(C_p, C_q)$ of any pair of clusters C_p and C_q, and IE *heaps* for maintaining the local/global minimum IE value in each merge.

Summary table is used to maintain the fast calculation of cluster entropy $\hat{H}(C_k)$ for each cluster (Figure 45). Since computing cluster entropy is based on counting the occurrences of each categorical value in each column, a summary table keeps these counters for the cluster. Apparently, if the average column cardinality is $|A|$, a summary table keeps $d|A|$ counters. Such a summary table enables fast merge operation – the two summary tables are added up to form the summary table for the new merged cluster.

We use IE**-table** to keep track of the incremental entropy between any pair of clusters, which is then used to maintain the minimum-IE with each cluster's IE **heap**s and a global IE heap, in each round. The IE-table is a symmetric table, where the cell (i, j) keeps the value of $IE(C_i, C_j)$ as shown in Figure 46. We define the most similar cluster of cluster u as

$$u.similar = \arg\min_v \{IE(u, v), v \neq u\}$$

Let $u.IE$ represent the corresponding incremental entropy by merging u and $u.similar$, and $< u, u.IE, u.similar >$ be the *feature vector* of cluster u. The $u.similar$ for each cluster u is maintained by a local heap, individually. The global minimum IE and the pair of clusters are maintained by the global heap of up-to-date feature vectors of all remaining clusters, sorted by $u.IE$.

After merging clusters, we need to update the bookkeeping information. Consider u as the main cluster in a merge, i.e., the most similar cluster $u.similar$ is merged to cluster u, we need to find a new $u.similar$ and put the new feature vector $< u, u.IE, u.similar >$ to the heap. Let v denote the old $u.similar$. The bookkeeping information for v is released and any entries in IE-table related to u or v should be updated, which counts for $O(N - k)$ incremental-entropy calculations at the round

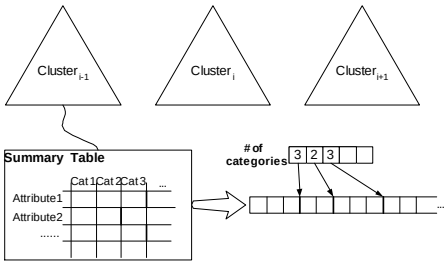

Figure 45: Summary table and physical structure

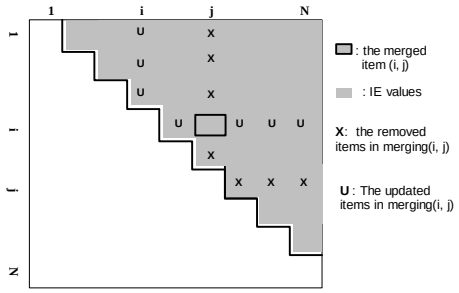

Figure 46: Illustration of the operation schedule after a merging operation

k. To maintain the most similar cluster of each cluster with the cluster's IE heap, a $O(\log(N-k))$ cost is necessary to update the list, which results in $O((N-k)\log(N-k))$ in total. For a few clusters having their most similar cluster changed due to the update of IE-table, their location at the heap needs to be updated, too. The update process is roughly illustrated by Figure 46.

4.6.2 Complexity of ACE

Updating the bookkeeping information is the most costly part, consisting of $O(N-k)$ incremental-entropy calculations and $O((N-k)\log(N-k))$ sorted list maintenance in the round k. Incremental-entropy calculation involves summing up the two summary tables and calculating the weighted entropies. The cost of computing weighted entropy can be minimized if we use an auxiliary array in length of N to store the $\log_2$ values. The following equation 15 shows that by using the $\log_2$ table we can calculate the weighted entropy with cost $O(d|A|)$, where $|A|$ is the average column cardinality.

$$n_p\hat{H}(C_p) = -\sum_{j=1}^{d} \sum_{\substack{v_{jk}\in A_j \\ c_{jk}=freq(v_{jk})|C_p}} n_p \frac{c_{jk}}{n_p}\log_2\frac{c_{jk}}{n_p} = -\sum_{j=1}^{d} \sum_{\substack{v_{jk}\in A_j \\ c_{jk}=freq(v_{jk})|C_p}} c_{jk}(\log_2 c_{jk} - \log_2 n_p)$$

Thus, the cost of N merge operations is $O(\sum_{k=1}^{N}\{(N-k)(d|A|+\log(N-k))\})$, i.e., $O(N^2\log N)$. In addition, the summary tables require $O(dN|A|)$ space, both the heaps and IE-table need $O(N^2)$ space. Therefore, when N becomes large, e.g., $N > 1000$, ACE algorithm cannot efficiently generate BKPlots. To address this problem, we discuss the properties of sample approximate BKPlots in next section. With the mean of sample BKPlots in appropriate sample size, we can still reliably find the Best Ks.

4.7 Sample BKPlots for Large Datasets

When the dataset is large (e.g., the number of records $N > 10,000$), ACE algorithm is not effective due to its complexity $O(N^2 \log N)$. There are two commonly accepted approaches to handle the large datasets: one is by sampling and the other is by summarization. We will focus on the sampling approach for finding the best Ks for large datasets, while the summarization approach has been reported for processing data streams [26].

In the sampling approach, we identify the best Ks based on the BKPlots generated on sample datasets, which we name it *sample BKPlots*, denoted by $B(n, K)$ at sample size n. We also name the BKPlot on the original dataset *original BKPlot*. Let S denote the number of sample BKPlots. We define the *mean BKPlot* of S sample BKPlots as follows.

$$E[B(n, K)] = \frac{1}{S} \sum_{i=1}^{S} B_i(n, K)$$

In order to prove that sample BKPlots can work effectively for large datasets, we need to answer three questions: 1) Do the mean BKPlots converge to the original BKPlot? 2)How is the quality of mean BKPlots in terms of consistency to the original BKPlot? 3) How to estimate the appropriate sample size that guarantees the mean BKPlot is consistent with the original BKPlot?

If mean BKPlots converge to the original BKPlot, we can confidently use mean BKPlots to estimate the original BKPlot. The quality of mean BKPlots should be related to the sample size and possibly other factors. We below first discuss the convergence of mean BKPlots which is used to evaluate the consistency of the mean BKPlots. Then we study the variance of mean BKPlots to evaluate the quality of the BKPlot estimation, and finally we develop the method for verifying whether a given sample size can guarantee a reliable mean BKPlot or not.

4.7.1 Convergence of mean BKPlots

In general, we model the clustering structure of a very large dataset $\mathbf{S}$ as follows. Let the primary clustering structure contains a few clusters, denoted as $C^M = \{C_1, C_2, \ldots, C_M\}$, e.g., $M = 20$. Assume each cluster is also very large, compared to the sample size n. Thus, uniformly sampling the large dataset to generate a meaningful sample dataset of n records is equivalent to proportionally sampling each of the M clusters and then composing the M sample clusters to form the sample

dataset. We denote the sample clusters with $C_n^M = \{C_{n,1}, C_{n,2}, \ldots, C_{n,M}\}$. With the definition of cluster entropy, we have the following proposition.

Proposition 6. *If the primary clustering structure is preserved with sample size n, $\hat{H}(C_{n,i})$ converges to $\hat{H}(C_i)$, when $n \to N$*

SKETCH PROOF. This proposition states that with the increasing sample size, the structure of sample cluster become more and more similar to that of original cluster. The proof can be described in three steps.

1) Sketch the distribution of any categorical value in a column of sample dataset. The cluster entropy $\hat{H}(C_i) = \sum_{j=1}^{d} \hat{H}(A_j|C_i)$ is defined by the probability p_{ijk} of each categorical value v_{jk} in column j, $1 \leq k \leq |A_j|$. Let N_i be the number of records in the cluster C_i, and N_{ijk} be the number of records containing the value v_{jk}. $p_{ijk} = \frac{N_{ijk}}{N_i}$. Let the random variable Y_{ijk} represent the number of records containing v_{jk} in sample cluster $C_{n,i}$. Y_{ijk} can be modeled as a *binomial distribution* $b(n_i, p_{ijk})$ [76], and thus $\frac{Y_{ijk}}{n_i}$ is an unbiased estimator for p_{ijk} with variance $\frac{p_{ijk}(1-p_{ijk})}{n_i}$, i.e., $E[\frac{Y_{ijk}}{n_i}] = p_{ijk}$ and $Var(\frac{Y_{ijk}}{n_i}) = \frac{p_{ijk}(1-p_{ijk})}{n_i}$.

2) We need to prove that the mean of $\frac{Y_{ijk}}{n_i} \log \frac{Y_{ijk}}{n_i}$ is $p_{ijk} \log p_{ijk}$. Let $Y = \frac{Y_{ijk}}{n_i}$. Using Taylor's formula [76], we have $Y \log Y = Y(\log p_{ijk} + c(Y - p_{ijk}))$, where c is some constant. We already know $E[Y] = p_{ijk}$ and $E[Y^2] = E^2[Y] + Var(Y) = \frac{p_{ijk}(1-p_{ijk})}{n_i} + p_{ijk}^2$. The mean of $Y \log Y$ is thus given by

$$\begin{aligned} E[Y \log Y] &= E[Y(\log p_{ijk} + c(Y - p_{ijk}))] = \log p_{ijk} E[Y] + cE[Y^2 - Y p_{ijk}] \\ &= p_{ijk} \log p_{ijk} + c(\frac{p_{ijk}(1 - p_{ijk})}{n_i}) \end{aligned}$$

When $n \to N$, the item $c(\frac{p_{ijk}(1-p_{ijk})}{n_i})$ becomes very small, so the right side converges to $p_{ijk} \log p_{ijk}$.

3) Since $H(A_j|C_{n,i}) = -\sum_{k=1}^{|A_j|} \frac{Y_{ijk}}{n_i} \log \frac{Y_{ijk}}{n_i}$, we have $E[H(A_j|C_{n,i})] \approx -\sum_{k=1}^{|A_j|} p_{ijk} \log p_{ijk} = H(A_j|C_i)$. It follows that $E[H(C_{n,i})] \approx H(C_i)$, when $n \to N$ □

Our ultimate goal is to study the mean and variance of the points on sample BKPlots, which are related to the entropy difference between the sample clustering structures, denoted as $I(n, K)$,

where n is the sample size and K is the number of cluster. Let a point on a sample BKPlot be $B(n, K)$. We show that we can use a set of sample BKPlots to estimate the original BKPlot for the entire large dataset as the following theorem.

Theorem 7. *The original BKPlot generated by the ACE algorithm for the large dataset can be estimated with the mean of sample BKPlots also generated by ACE.*

SKETCH PROOF. The proof consists of two steps.

1) $E[I(n, K)]$ converges to the approximate $I(K)$ generated by the ACE algorithm. When the ACE algorithm is used to generate BKPlots, the entropy difference $I(K)$ between the nearby clustering schemes is $I(K) \approx \frac{1}{N} IE^{(K)} = \frac{1}{N}\{(N_p + N_q)\hat{H}(C_p + C_q) - N_p\hat{H}(C_p) - N_q\hat{H}(C_q)\}$. Similarly, it applies to the sample datasets, i.e., $I(n, K) \approx \frac{1}{n}\{(n_p+n_q)\hat{H}(C_{n,p}+ C_{n,q}) - n_p\hat{H}(C_{n,p}) - n_q\hat{H}(C_{n,q})\}$. Since $E[\hat{H}(C_{n,i})]$ converges to $\hat{H}(C_i)$, and $N_p/N \approx n_p/n$, $N_q/N \approx n_q/n$, $E[I(n, K)]$ also converges to the approximate $I(X)$.

2) Since the BKPlot is based on the $I(K)$ curve, i.e., $B(n, K) = I(n, K - 1) - 2I(n, K) - I(n, K + 1)$, the mean of sample BKPlots $E[B(n, K)]$ will also converge to the original BKPlot $B(K)$ generated by the ACE algorithm. □

How reliable $E[B(n, K)]$ can be used to represent the original BKPlot depends on its variance property. Before we design the method to testing the reliability of a mean BKPlot, we first study the variance of mean BKPlots.

4.7.2 Variance of mean BKPlots

The variance of mean BKPlot, $Var(E[B(n, K)])$, can be used to evaluate the quality of BKPlot estimation. We are more interested in the asymptotic variance, especially the relationship between variance and sample size. We will study the asymptotic variance of $E[B(n, K)]$ in four steps. 1) We derive the general form of asymptotic variance of the sum of random variables: $Var(\sum_{i=1}^{m} a_i X_i)$, which will be heavily used in deriving other results; Then, we show that 2) $Var(E[\hat{H}(C_{n,i})]) \sim O(\frac{1}{n_i S})$; 3)$Var(E[I(n, K)]) \sim O(\frac{1}{n_i S})$; and 4) $Var(E[B(n, K)]) \sim O(\frac{1}{n_i S})$.

1) We will repeatedly use the following formula in the estimation. Let X_i denote the ith random variable, $i = 1 \ldots m$, $Var(X_i) \sim O(\frac{1}{n_i})$, where n_i is the sample size of X_i, a_i are some

constants, and X_i and X_j are correlated with correlation coefficient ρ_{ij}, $|\rho_{ij}| \leq 1$. For simplicity, we assume that ρ_{ij} is approximately a constant, and is not related to the sample size n_i.

$$\begin{aligned} Var(\sum_{i=1}^{m} a_i X_i) &= \sum_{i=1}^{m} a_i^2 Var(X_i) + 2 \sum_{i<j\leq m} a_i a_j \rho_{ij} \sqrt{Var(X_i)Var(X_j)} \\ &\sim O(\sum_{i=1}^{m} \frac{a_i^2}{n_i} + 2 \sum_{i<j\leq m} \frac{a_i a_j}{\sqrt{n_i n_j}}) \end{aligned} \tag{15}$$

When $n_i \equiv n$, the result is simplified to $O(\frac{\sum_{i=1}^{m} a_i^2 + (\sum_{i=1}^{m} a_i)^2}{n}) \sim O(\frac{1}{n})$

2) Similar to the process of getting $E[Y \log Y]$, we apply Taylor's formula [76] to expand $Y \log Y$, then apply the formula (15) to get $Var(Y \log Y) \sim O(\frac{1}{n} + \frac{1}{n^2}) \sim O(\frac{1}{n})$. With the Central Limit Theory [76], suppose there are S sample BKPlots, the distribution of $E[\frac{Y_{ijk}}{n_i} \log \frac{Y_{ijk}}{n_i}]$ can be approximated with a normal distribution $N(p_{ijk} \log p_{ijk}, O(\frac{1}{n_i S}))$, where $O(\frac{1}{n_i S})$ is the asymptotic notation of the variance. By the definition of column entropy and applying the formula (15), we have $Var(E[\hat{H}(A_j|C_i)]) \sim O(\frac{f(|A_j|)}{n_i S})$, where $f(|A_j|)$ is some constant determined by $|A_j|$. Let ρ'_{jk} be the correlation between $E[\hat{H}(A_j|C_i)]$ and $E[\hat{H}(A_k|C_i)]$, from the definition of adjusted entropy, we have

$$\begin{aligned} Var(E[\hat{H}(C_{n,i})]) &= Var(\sum_{j=1}^{d} \frac{1}{d \log |A_j|} E[\hat{H}(A_j|C_i)]) = \sum_{j=1}^{d} \frac{1}{d^2 \log^2 |A_j|} Var(E[\hat{H}(A_j|C_i)]) \\ &+ 2 \sum_{j<k\leq d} \frac{1}{d^2 \log |A_j| \log |A_k|} \rho'_{jk} \sqrt{Var(E[\hat{H}(A_j|C_i)])Var(E[\hat{H}(A_k|C_i)])} \\ &\sim O(\sum_{j=1}^{d} \frac{f(|A_j|)}{d^2 \log^2 |A_j| n_i S}) \sim O(\frac{1}{n_i S}) \end{aligned} \tag{16}$$

3) Since we suppose $I(n, K) \approx \frac{1}{n} IE(C_p, C_q)$ with ACE algorithm, we can estimate the variance for $E[I(n, K)]$ as follows.

$$\begin{aligned} Var(E[I(n,K)]) &= Var\{\frac{n_p + n_q}{n} E[\hat{H}(C_p + C_q)] + \frac{n_p}{n} E[\hat{H}(C_p)] + \frac{n_q}{n} E[\hat{H}(C_q)]\} \\ &\sim O(\frac{n_p + n_q}{n^2 S} + \frac{n_p}{n^2 S} + \frac{n_q}{n^2 S} + 2(\frac{\sqrt{(n_p + n_q)n_p}}{n^2 S} + \frac{\sqrt{(n_p + n_q)n_q}}{n^2 S} + \frac{\sqrt{n_p n_q}}{n^2 S})) \\ &\sim O(\frac{n_p + n_q}{n^2 S}) \end{aligned} \tag{17}$$

To simplify it further, the variance is asymptotically $O(\frac{1}{nS})$ for $n_p + n_q \sim O(n)$.

4) By definition of $B(K)$, we have $B(n,K) = I(n,K-1) - 2I(n,K) - I(n,K+1)$, which is a linear combination of the entropy difference between partition schemes. With formula 15, we have $Var(E[B(n,K)])$ as

$$Var(E[B(n,K)]) = Var(B(n,K))/S \sim O(\frac{1}{nS})$$

Therefore, by increasing the sample size, n, or the number of sample sets, S, the mean BKPlot will be more reliable.

Now we show how the mean BkPlots relate with the number S of sample sets and the sample size n with the real census dataset. Readers may refer to Section 4.9 for a detailed description about the dataset. For each sample size ($n = 200 \sim 1000$), we generate 10 sample BKPlots. Figure 47 shows that although the sample BKPlots might have larger variance, the resulting mean BKPlots are very close. Figure 48 focuses on the peak values at K=3 in Figure 47. Figure 49 shows that the sample variance decreases with the increasing sample size. For the same sample size, a smaller K involves merging larger clusters, i.e., larger $n_p + n_q$ in Formula (17), which results in a slightly larger variance consistent with the Formula 17.

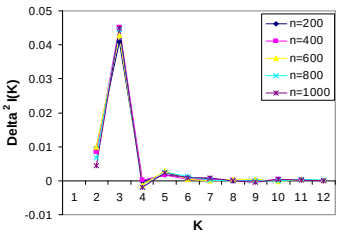

Figure 47: Comparing the mean BKPlots at different sample size

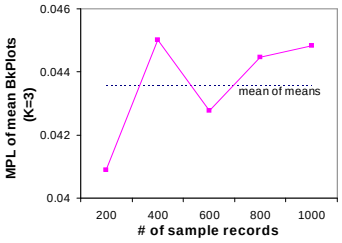

Figure 48: The most significant values on the mean BKPlots (K=3)

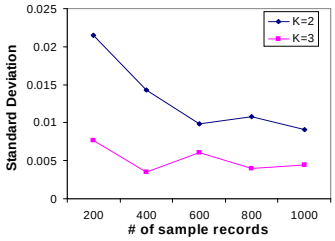

Figure 49: The variance of the sample BKPlots at K=2 and K=3

4.7.3 Conditions for a Reliable Mean BKPlot

In general, the larger n and S, the smaller the variance of mean BKPlots, which results in more reliable estimation. However, in practice, we want to use limited sample size (n) and limited number of sample BKPlots (S) to get a reliable mean BKPlot that is consistent with the original BKPlot. In last section, we have found the relationship between the variance and the sample factors, with which we are able to verify if the sample datasets can generate a reliable mean BKPlot.

We first define the concept of consistent mean BKPlot. Suppose that the top κ number of candidate Ks (e.g., $\kappa = 3$ and $K < 20$) on the original BKPlot have $B(K)$ values *significantly* higher than certain level η. Without loss of generality, let $k_1 < k_2 < \ldots < k_\kappa$ be the significant Ks on the original BKPlot ordered by $B(k_i)$, $1 \leq i \leq \kappa$.

Definition 13. *If the sequence of significant Ks on the mean BKPlot $k'_1 < k'_2 < \ldots < k'_\kappa$ satisfies $k'_i = k_i$ for $1 \leq i \leq \kappa$, the mean BKPlot is consistent with the original BKPlot.*

There are $\kappa + 1$ significant levels on the original BKPlot: $B(k_i)$, $1 \leq i \leq \kappa$, and η. Let the minimum difference between the $\kappa + 1$ values be Δ by certain $B(k_r)$ and $B(k_s)$, $r < s$, i.e., $\Delta = |B(k_r) - B(k_s)|$. With Central Limit Theorem, $E[B(n, K)]$ follows a normal distribution with mean $B(K)$ and variance $Var(B(n, K))/S$. Let c be the constant related to certain confidence level [76], for example, $c = 1.96$ for $CL = 95\%$ confidence level. In order to make the κ levels consistent with the original BKPlot, we should guarantee the non-overlapping confidence intervals of $B(n, k'_i)$ at the significant Ks, with confidence CL, i.e., the following constraint is satisfied.

$$c(\sqrt{Var(B(n, k'_i))/S} + \sqrt{Var(B(n, k'_j))/S}) = 2c\sqrt{Var(B(n, K))/S} < \Delta, 1 \leq i \leq \kappa$$

It follows that, if the sample variance at size n satisfies the following formula, the mean BKPlot will be consistent with the original BKPlot.

$$Var(B(n, K)) < \frac{\Delta^2 S}{4c^2} \tag{18}$$

This provides a testing method for the reliability of the mean BKPlot in term of sample size n, the number of sample BKPlots S, and the interested top Ks. We can start with some initial n and S for the given dataset, e.g., $n = 1000$ and $S = 10$. Then, we get the estimation of $Var(B(n, K))$ and Δ from the sample BKPlots. If the Formula (18) is satisfied with the estimated $Var(B(n, K))$ and Δ, we say that the mean of sample BKPlots is reliable under certain confidence level CL. Otherwise, we can increase either n, or S. Obviously, if $Var(B(n, K))$ is not so large, increasing S is more economic.

4.8 Identifying No-cluster Datasets with BKPlot Method

Finding the peaks on BKPlot does not always mean that there is significant clustering structure. Low peak levels mean smooth changes of clustering structure and each clustering structure does not

distinguish itself from others, i.e., there is no significant clustering structure. Therefore, a challenge comes – how to distinguish datasets having clustering structures from those having no clustering structure? We propose a method to address this problem. The main idea is based on the property of the BKPlots of the sample datasets that are known to have *no* clustering structure. Specifically, first, we study the noisy clustering structures in the typical no-cluster datasets: datasets with uniform or normal data distribution (single mode). Then, we test if the clustering structure of the given dataset is significantly different from the noisy no-cluster structures.

4.8.1 Property of Datasets Having No Clustering Structure

We observe two types of typical datasets that do not have significant clustering structure. One is the datasets with elements uniformly distributed within a column and between columns, which obviously have no clustering structure. The other type is the datasets of discretized multidimensional normal distribution, which has only one mode (or one cluster). In the following discussion, the Maximum Peak Level (MPL) of BKPlot is used to represent the significance level of the clustering structure.

Ideally, large sample datasets exactly following uniform/normal distribution have no significant clustering structure. Thus, every point on their BKPlots should be zero or very close to zero. However, synthetic sample datasets usually do not exactly follow the desired distribution, which may create some small noisy structures. These noisy structures should be treated as statistically equivalent to the ideal no-cluster structures. Any datasets that have clustering structures not significantly different from these noisy structures should be treated as no-cluster datasets, too. With the varying number of records n, columns d, and column cardinalities $|A_j|$, $1 \leq j \leq d$, synthetic sample datasets may deviate from the desired distribution to different degrees, thus create different levels of noisy structures. We shall characterize them both formally and experimentally.

Formally, from Formula (16), we can get a more detailed form of the $Var(E[B(n,K)])$, in terms of the three factors n, d, and $|A_j|$.

$$Var(E[B(n,K)]) \sim O(\sum_{j=1}^{d} \frac{f(|A_j|)}{d^2 \log^2 |A_j| nS}) \tag{19}$$

For ideal no-cluster datasets, $B(K) \approx 0$ everywhere. Thus, with the increasing n or d, the MPLs should converge to 0. The converging rates are characterized by the corresponding factors in

$Var(E[B(n,K)])$. From above formula we are unable to determine the effect of $|A_j|$ yet. We will show some experimental results to further study the relationship between the three factors and MPLs.

4.8.2 Testing Existence of Significant Clustering Structure

From the above analysis, we conclude that different settings of n, d and $|A|$ may result in different statistical properties of no-cluster BKPlots. There is no simple threshold good for all kinds of datasets. We have to characterize the statistical property of the no-cluster BKPlots in terms of the specific setting of the three factors, i.e., given a real dataset, its sample BKPlots have to be compared to those of no-cluster datasets that have the same setting of n, d, and $|A_j|$.

Combined with the theory of sample BKPlots, we suggest the following statistical testing method for determining the significance of the clustering structure in a given dataset.

1. generate the BKPlot with ACE algorithm for the target dataset and find its MPL denoted by μ';

2. with the n, d and $|A_j|$ setting of the target dataset, we generate two sets of testing datasets : one with uniform distribution and the other with normal distribution, each with about 30 datasets [1];

3. calculate the mean denoted by μ, and the confidence interval of the MPLs of the simulated datasets: $[\mu - CI, \mu + CI]$, at confidence level λ;

4. if $\mu' > \mu + CI$, there is significant clustering structure in the target dataset, otherwise, no clustering structure. CI is often very small compared to the mean level μ, thus, μ is sufficient to represent the upper bound.

4.9 Experiments

We have formally given the basic BKPlot method, the sample BKPlot method for large datasets, and the BKPlot method for identifying no-cluster datasets. In this section, we want to show that, 1) BKPlots can be used to effectively find the critical Ks; 2) experimental results support the initial

[1] a number which is considered as "statistically large"

analysis of the noisy structures of typical no-cluster datasets; 3) ACE algorithm is a robust tool for generating high-quality approximate BKPlots, compared to the existing entropy-based clustering algorithms, represented by Monte-Carlo method (MC) [78], Coolcat [12], and LIMBO [7].

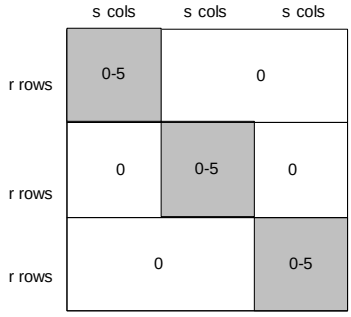

Figure 50: Synthetic Data DS1

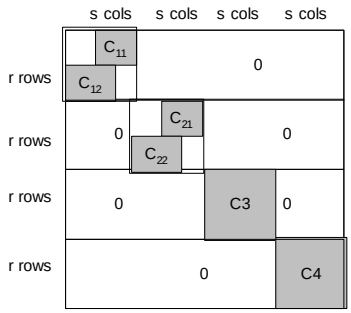

Figure 51: Synthetic Data DS2

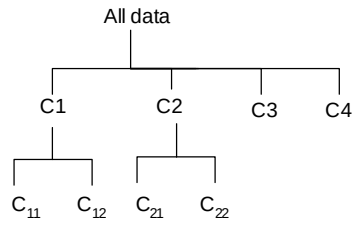

Figure 52: The significant clustering structures in DS2

4.9.1 Datasets

In order to intuitively evaluate the effectiveness of the method, we use both synthetic and real datasets, and cross-validate the results with a visualization tool VISTA [22] that is designed for validating numerical clustering results.

Simple Structured Synthetic Data (DS1&DS2). First, we construct two sets of synthetic categorical datasets, so that the clustering structure can be intuitively understood and verified. The first set of datasets have a multi-layered clustering structure, which can be visually verified. Figure 51 shows such a dataset with 1000 records and 30 columns. It has a double-layered clustering structure. The top layer has four clusters, two of which have two sub-clusters, respectively. Each cluster has random categorical values selected from {‘0’,‘1’,‘2’,‘3’,‘4’, ‘5’} in a distinct set of attributes, while the rest attributes are set to ‘0’. As we can visually identify it, its BKPlot should at least suggest two Best Ks. Similarly, a single-layered structure is shown as Figure 50. We name these single-layered and double-layered structure as DS2 and DS1, respectively.

Cross-validated Mixture Data (DS3). The third set of datasets, are the discretized version of multidimensional normal mixture datasets [57]. We use this set of datasets because we can cross-validate the result with both BKPlot method and the visualization method designed only for numerical data. As we discretize the continuous value while preserving the numerical meaning, the

original numerical clusters are still preserved. Therefore, we can still use the visualization system VISTA [22] to visually validate the clusters, when we can also use the BKPlot method to find the best Ks. A discretized 1K-record 10-dimensional mixture dataset with five clusters is generated for the cross-validation purpose (Figure 53). The continuous values in each column are partitioned into 10 equal-width buckets for discretization, i.e., the discretized dataset has column cardinality of 10 for each column. The 5 clusters can be further grouped, such as C1.1 vs. C1.2, and C2.1 vs. C2.2, to form a two-layer clustering structure.

Census Data. We also use a real dataset – the discretized version of the large 1990 census data [2]. This dataset is originally used by the paper [82] in studying the relationship between the sampling approach and the effectiveness of Expectation-Maximization (EM) based clustering algorithms for very large datasets. It is very large in terms of both the number of records and the number of attributes. After dropping many of the less useful attributes in the raw dataset, the total number of preserved attributes still reaches 68. It contains more than 2 million (2,458,284) records, about 352 megabytes in total. Since the discretized version still preserves the distance meaning, we also expect that the BKPlot method will be consistent with the visual validation method, if the BKPlot method is effective.

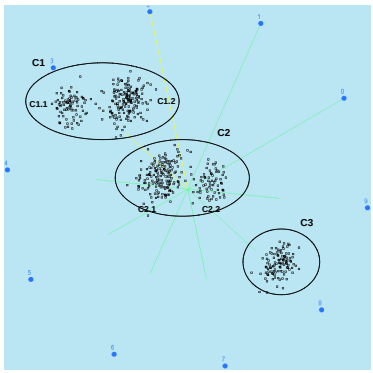

Figure 53: Visualization of the discretized mixture data DS3.

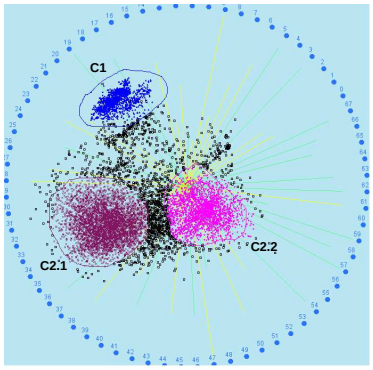

Figure 54: Visualization of 10K-record sample census data, with dense areas manually labeled

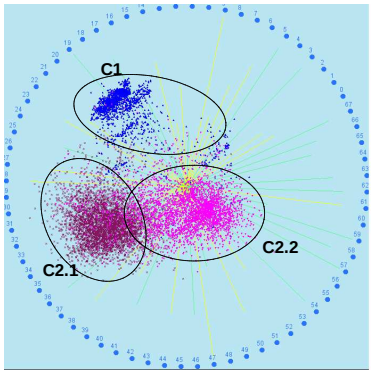

Figure 55: Sample census data labeled by ACE algorithm.

[2] In UCI KDD Archive http://kdd.ics.uci.edu/databases/census1990/USCensus1990.html

4.9.2 Validating the BKPlot Method

Below we show the result of applying the BKPlot Method to the synthetic and real datasets. The ACE algorithm is used to generate the BKPlots. We generate ten sample datasets for each type of synthetic clustering structure.

DS1. The BKPlots generated by ACE algorithm for DS1-i datasets (Figure 56 clearly indicate that '3' is the only significant K and datasets with the same clustering structure have almost the identical BKPlot. By checking the significant level for the setting of 1000 records, 30 columns and column cardinality of 6, with 20 no-cluster test datasets (10 normal distribution datasets and 10 uniform distribution datasets, respectively), we find the maximum peak levels (MPLs) of no-cluster datasets are around 0.0004, which is far lower than that of the DS1, 0.08. Therefore, the detected Best K is significant.

DS2. The peaks of BKPlots for DS2-i (Figure 57) include the two inherent significant Ks – '4' and '6'. However, '2' is also given as the third significant K, which suggests that the top 4 clusters can be further clustered into two groups. Interestingly, compared to the three peak levels, we notice that the peak values at 'K=2' have much higher variance, which implies that 'K=2' is less significant than the other two.

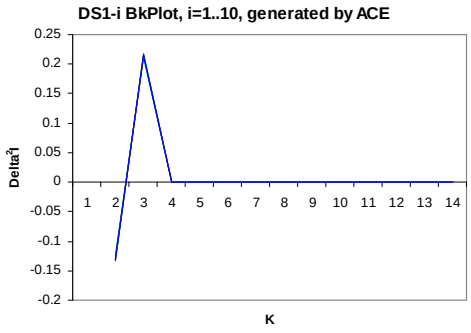

Figure 56: BKPlots of DS1 by ACE

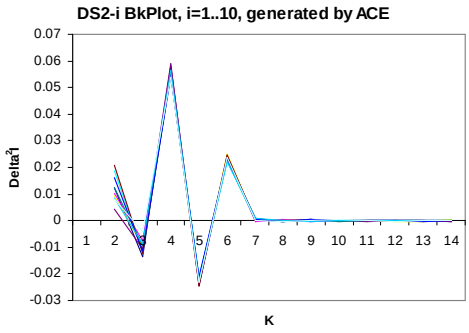

Figure 57: BKPlots of DS2 by ACE

DS3. The BKPlot generated by ACE algorithm (Figure 58) indicates exactly that 5 and 3 are the Best Ks, and they are significant compared to the bound ($\sim$0.013) for the setting of 1K records, 10 columns and column cardinality of 10. The corresponding labeling result shows the clustering/validation result is well matched with the visualization result (Figure 53).

Census data. The Census data contains three major clusters as shown in the visualization of

1K sample dataset (Figure 54. C2 and C3 are very close, thus, they may also form a two-layer clustering structure $-$ the top layer consists of C1 and C2+C3. Its BKPlot (Figure 59 indicates both 2 and 3 are significant, compared to the bound ($\sim$ 0.0005) for the setting (1000 records, 68 columns and column cardinality defined in [82]). The clustering result with ACE at $K = 3$ is very close to the cluster distribution observed via visualization (Figure 55). The cross-validation with the visualization method confirms that the Best K and clustering result generated by ACE algorithm are highly consistent with the inherent clustering structure.

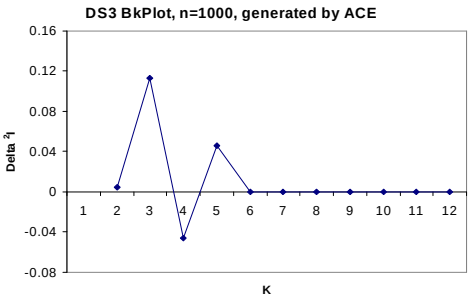

Figure 58: BKPlots of DS3 by ACE

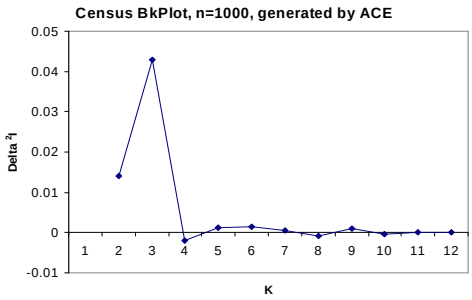

Figure 59: BKPlots of census dataset by ACE

We summarize the results in Table 6, where n represents the number of records used in generating BKPlots, d is the number of columns, and "Cardinality" is the column cardinality, i.e., $|A|$. For the first three datasets, each column has the same cardinality. For census dataset, the column cardinality varies from 2 to 223. Clustering structure describes the possible hierarchical structure of dataset. For example, "two layers, 4/6 clusters" for DS2 means that the clustering structure has two layers with 4 and 6 clusters, respectively. "MPL bounds" is the estimated upper bound of no-cluster datasets with the same setting of n, d and $|A|$. Due to very small confidence intervals, only the mean levels are used to represent the bounds.

Table 6: Summary of experiments for the BKPlot method

Datasets	n	d	cardinality	Clustering Structure	MPL bounds	BestK	MPLs at Best Ks
DS1	1000	30	6	single layer, 3 clusters	0.0004	3	0.082
DS2	1000	30	6	two layers, 4/6 clusters	0.0004	2, 4, 6	0.005, 0.023, 0.010
DS3	1000	10	10	two layers, 3/5 clusters	0.013	3,5	0.113,0.046
Census sample	1000	68	2-223	two layers, 2/3 clusters	0.0005	2,3	0.014,0.044

4.9.3 BKPlot vs. BIC

Bayesian Information Criterion (BIC) [57] is a popular method for model selection. If appropriate assumption is made about the prior distribution of the clusters, it can also be used to select the best K number of clusters. We compare our method with BIC and show the unique advantages of our method.

In order to use BIC method, we need to make assumption about the cluster distribution. Categorical data is often modeled with Multinomial mixture [76]. The model fitting is optimized by maximizing the likelihood of fitting the data to the mixture model. The generic form of BIC is then based on the maximum likelihood.

$$BIC = -2 \cdot \log \text{likelihood} + (\log n) \cdot \psi$$

where n is the number of sample records, and ψ is the number of parameters used in the modeling, including the number of clusters. Since only the number of clusters changes in determining the best Ks, we can use $BIC = -2 \cdot \log likelihood + (\log n) \cdot K$ instead, where likelihood is the maximum likelihood of the K-cluster mixture model. The main problem is, if the real cluster distribution does not well follow the assumed distribution with any K, the result is possibly not good. For example, in numerical clustering, the Gaussian mixture model does not work well for irregularly shaped clusters [22].

In experiments, we use AutoClass[3] to generate the fitted model. On the BIC curve, the best K happens at the knee where the BIC stops dropping dramatically. We observed that the BIC method can often suggest one best K, but it cannot find all possible best Ks for multi-layer clustering structures. In the experiment we also observed, while BIC clearly suggests one of the two best Ks for DS2 and DS3 (Figure 60 and 61), the best K for Census data is not clearly indicated (Figure 62) because of the overlapping clusters and the outliers.

4.9.4 Experiments on Datasets Having no Clustering Structure

In this set of experiments, we want to show the relationship between the Maximum Peak Levels (MPLs) of no-cluster sample datasets and the factors: the number of records n, the number of

[3]http://ic.arc.nasa.gov/ic/ projects/bayes-group/autoclass/

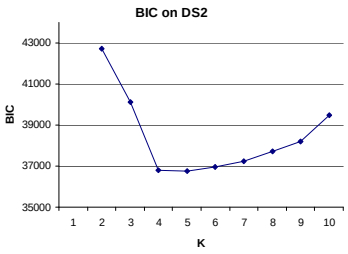

Figure 60: BIC suggests only K=4 for DS2

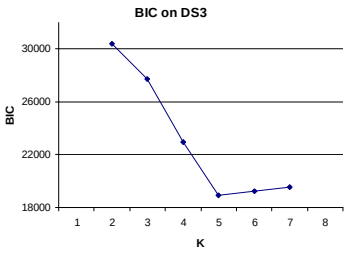

Figure 61: BIC suggests only K=5 for DS3

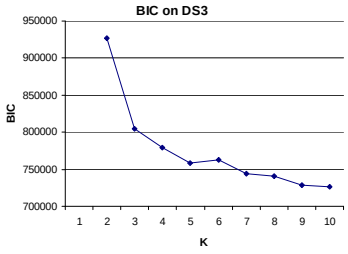

Figure 62: BIC probably suggests K=3 for Census data

columns d, and the column cardinalities $|A_j|$, which should be consistent with our formal analysis.

Since n and d have similar effect on the sample BKPlots according to the analysis, we organize them in the first set of experiments, while the second set of experiments focus on the unknown effect of column cardinality. Each point on the figures is the average of ten runs with the standard deviation as the error bar.

Number of Records and Number of Columns

For simplicity, the set of simulated datasets in this experiment have the equal cardinality for columns, denoted as $|A|$ – with unequal cardinality, we can get similar results. Figure 63 shows the result when we fix the number of columns and the column cardinality, and vary the number of records only. The MPL drops quickly from the sample size 100 to 300, but keeps stable when the size increases more. This confirms our analysis that for small datasets, the entropy differences between the clusters have large variance and MPLs tend to deviate more from zero. Varying the number of columns, while fixing the other two factors, we get a similar pattern, as Figure 64 shows.

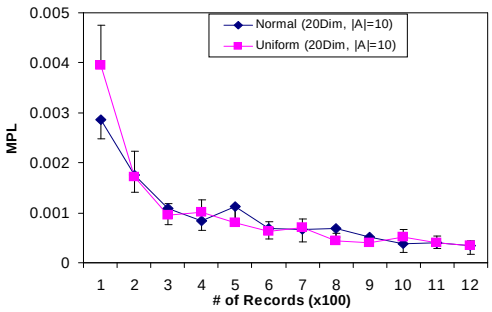

Figure 63: Relationship between the number of records and MPLs

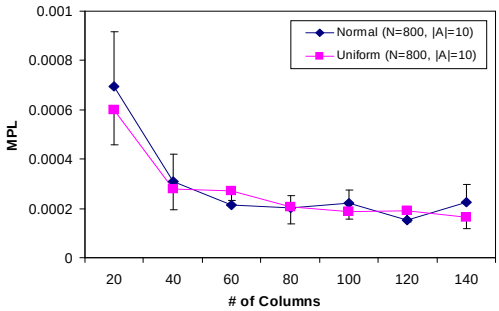

Figure 64: Relationship between the number of columns and MPLs

Column Cardinality

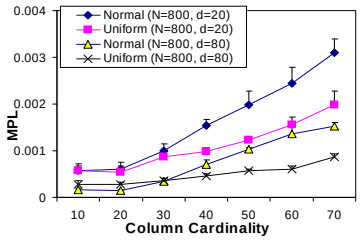

Figure 65: Relationship between the column cardinality and MPLs

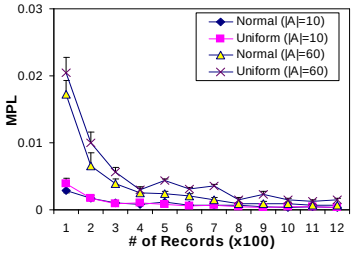

Figure 66: The number of records vs. MPLs with two levels of column cardnality

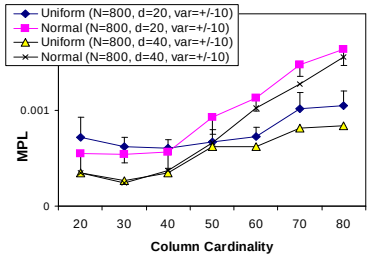

Figure 67: unequal column cardinality vs. MPLs

We can try to understand the factor of cardinality in terms of the model complexity – column cardinality represents the inherent complexity of dataset. In general, with the increasing model complexity, the representative sample size should be increased in order to capture the complexity of the structure. On the other hand, if the sample size n keeps unchanged the increasing model complexity should bring more variance, which results in stronger noisy structures, i.e., higher MPLs. This hypothesis is supported by the experiments (Figure 65) that increasing the mean column cardinality will increase the level of MPLs with the same sample size. Figure 66 also shows that with the increasing sample size, the higher the column cardinality is, the slower the MPLs converge for both types of datasets (uniform and normal), which confirms that we need more samples to characterize the increasing model complexity. In Figure 67, we let the column cardinalities randomly varying in the range $[|A| - 10, |A| + 10]$, but keep the mean cardinality as $|A|$. We also get a similar pattern that the MPLs increase when the mean cardinality increases.

4.9.5 Other Algorithms for Generating Approximate BKPlots

Literally, any categorical clustering algorithm that employs the same entropy minimization criterion can generate approximate BKPlots. However, the quality of approximate BKPlots can be greatly influenced by the algorithms. We compare three directly related algorithms: Monte-Carlo [78], Coolcat [12], and LIMBO [7] algorithm in this section. Monte-Carlo and Coolcat use the same criterion, "expected-entropy", that is also used by ACE, to find suboptimal partitions, while LIMBO uses mutual information in clustering. We also modify the first two algorithms to use the adjusted expected-entropy defined in section 4.4. The reported results are based on ten sample datasets for each experimental data.

4.9.5.1 Algorithms

Monte-Carlo Method [78] is a top-down partitioning algorithm. With a fixed K, it begins with all records in one cluster and follows an iterative process. In each step, the algorithm randomly picks one record from one of the K clusters and puts it into another randomly selected cluster. If the change of assignment does not reduce the expected entropy, the record is put back to the original cluster. Theoretically, given a sufficiently large s, the algorithm will eventually terminate at a near optimal solution. We set $s = 5000$ for running MC on the synthetic datasets.

Coolcat [12] algorithm begins with selecting K records, which maximize the K-record entropy, from a sample of the dataset as the initial K clusters. It sequentially processes the rest records and assigns each to one of the K cluster. In each step, the algorithm finds the best fitted one of the K clusters for the new record – adding the new record to the cluster will result in minimum increase of expected entropy. The data records are processed in batches. Because the order of processing points has a significant impact on the quality of final clusters, there is a "re-clustering" procedure at the end of each batch. This procedure picks m percentage of the worst fitted records in the batch and re-assigns them to the K clusters in order to reduce the expected entropy further.

We run Coolcat algorithm on each dataset with a large initial sample size (50% of the dataset) for choosing the seed clusters and $m = 20\%$ for re-clustering, which is sufficient for improvement through re-clustering [12]. In order to reduce the effect of ordering, we also run Coolcat 20 times for each datasets and each run processes the data in a randomly generated sequence. Finally, we select the result having the lowest expected entropy among the 20 results.

LIMBO [7] algorithm is a hierarchical clustering algorithm using the Information Bottleneck [101] criterion as the similarity between clusters. It uses a summarization structure DCF-tree to condense large datasets. In our experiments, we set the information loss factor Φ to 0, which does not use DCF-tree to compress the data. Under this setting the result is not subject to the order of records, and thus there is no randomness introduced in different runs for the same dataset.

4.9.5.2 Measures for Quality of BKPlots

We use three measures to evaluate the quality of approximate BKPlots.

- *Coverage Rate.* The robustness of BKPlot is represented with "Coverage Rate (CR)" – how

many significant inherent clustering structures are indicated by the BKPlot. There could be more than one significant clustering structures for a particular dataset. For example, four-cluster and six-cluster structures can be all significant for DS2. An robust BKPlot should always include all of the significant Ks.

- *False Discovery Rate.* There could be some Ks, which are actually not critical but suggested by some BKPlots. In order to efficiently find the most significant ones, we prefer a BKPlot to have less false indicators as possible. We use "*False Discovery Rate*(FDR)" to represent the percentage of the noisy results in the BKPlot.

- *Expected Entropy.* Since the BKPlot is indirectly related to expected entropy, it is also reasonable to check the quality of expected entropy for the partitions generated by different algorithms at the particular Ks. For a set of datasets in the same clustering structure, like DS1-i, $1 \leqslant i \leqslant 10$, we have almost same optimal clustering structure for different datasets at a fixed K. Using the mean-square-error (MSE) criterion [76] to evaluate the quality of the algorithmic result, we can decompose the errors to two parts: the deviation to the lowest expected entropy (the expected entropy of the optimal partition), and the variance of the estimated expected entropy. Let $\hat{h}$ be the expected entropy of the clustering result and h be the optimal one. $\hat{h} \geq h$ is held. Let $E[\hat{h} - h]$ be the expected bias and $var(\hat{h})$ is the variance of $\hat{h}$.

  $$MSE = E^2[\hat{h} - h] + var(\hat{h})$$

  Without knowing h, if an algorithm generates clustering results with the lowest expected entropy and minimum variance among other algorithms, its BKPlots might be more trustable.

4.9.5.3 Results by Other Algorithms

As we have shown in Section 4.9.2, the BKPlots generated by ACE algorithm clearly and consistently indicate the exact Best Ks for the experimental datasets. We show some results generated by other algorithms.

The BKPlots generated by Monte-Carlo algorithm for DS1 (Figure 68) clearly identify that '3' is the best K with some small variation. However, BKPlots for DS2 show large variation on Ks.

Overall, the Ks distribute from '2' to '10' for different DS2-i, not allowing the user to identify the exact Best Ks. This implies that MC algorithm might not be robust enough for datasets having complicated clustering structure. The reason is MC algorithm becomes more likely to trap in local minima with the increasing complexity of clustering structure and increasing number of clusters.

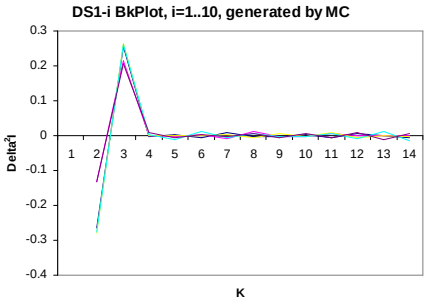

Figure 68: BKPlots of DS1 by MC

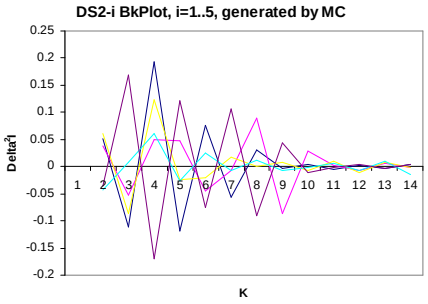

Figure 69: BKPlots of DS2 by MC

Coolcat algorithm is even worse. It brings large variation for both datasets (Figure 70 and 71). The reason is that Coolcat algorithm simply does not get the near-optimal clustering result for any fixed number of clusters. Therefore, it is not suitable for generating robust BKPlots.

LIMBO can successfully find the best Ks for simple structures, such as DS1 and DS2. Its DS1 and DS2 BKPlots 72 are very similar to ACE's. However, it may generate some noise and miss some best Ks for a more complicated structure, such as Census data 73. The LIMBO result can give the top best K at $K = 3$, while it misses the higher level $K = 2$ and also have some noise at $K = 5$.

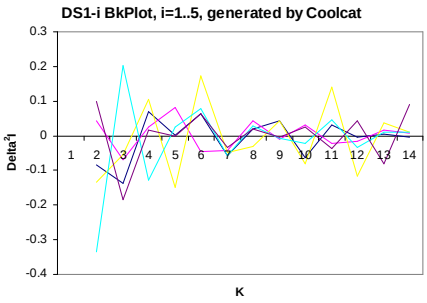

Figure 70: BKPlots of DS1 by Coolcat

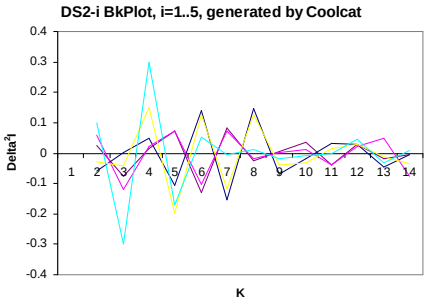

Figure 71: BKPlots of DS2 by Coolcat

We summarize the results with the discussed measures, Coverage Rate (CR), False Discovery Rate (FDR), and expected entropy (EE) in Table 7. The higher the coverage rate, the more robust the BKPlot is. The lower the false discovery rate the more efficient the BKPlot is. The numbers

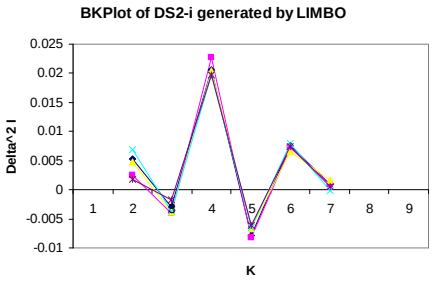

Figure 72: BKPlots of DS2 by LIMBO

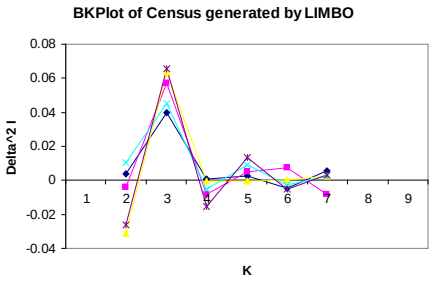

Figure 73: BKPlots of Census data by LIMBO

Table 7: Quality of Approximate BKPlots

		CR	FDR	EE(K=3)	
DS1	ACE	100%	0%	0.283 ± 0.000	
	LIMBO	100%	0%	0.283 ± 0.001	
	MC	100%	0%	0.283 ± 0.001	
	Coolcat	60%	85%	0.285 ± 0.005	
		CR	FDR	EE($K=4$)	EE($K=6$)
DS2	ACE	100%	33%	0.218 ± 0.001	0.194 ± 0.001
	LIMBO	100%	33%	0.218 ± 0.002	0.194 ± 0.001
	MC	80%	53%	0.244 ± 0.015	0.214 ± 0.005
	Coolcat	60%	70%	0.253 ± 0.005	0.216 ± 0.008
		CR	FDR	EE($K=3$)	EE($K=5$)
DS3	ACE	100%	0%	0.346 ± 0.003	0.249 ± 0.003
	LIMBO	100%	0%	0.349 ± 0.004	0.250 ± 0.005
	MC	100%	0%	0.346 ± 0.003	0.248 ± 0.005
	Coolcat	80%	29%	0.350 ± 0.004	0.275 ± 0.024
		CR	FDR	EE($K=2$)	EE($K=3$)
Census	ACE	100%	0%	0.398 ± 0.004	0.336 ± 0.004
	LIMBO	60%	33%	0.416 ± 0.010	0.338 ± 0.012
	MC	50%	17%	0.395 ± 0.004	0.330 ± 0.007
	Coolcat	50%	17%	0.401 ± 0.003	0.341 ± 0.007

are the average over the 10 sample datasets. In almost all cases, ACE shows the minimum expected entropy and minimum standard deviation, as well as the highest CR and lowest FDR. LIMBO is the second reliable method for generating approximate BKPlots. In general, it can generate reliable BKPlots for not-so-noisy and non-overlapping clustering structure. MC algorithm can possibly find the best clustering result in some cases, such as DS3 and census data.

CHAPTER V

RANDOM GEOMETRIC PERTURBATION APPROACH TO PRIVACY PRESERVING DATA CLASSIFICATION

The iVIBRATE framework is a novel application of geometric methods in data clustering. This chapter presents a random geometric perturbation approach to privacy preserving data classification. The goal of this approach is two-fold: preserving the accuracy of classifiers and preserving the privacy of data. To achieve the first goal, we identify that many classification models utilize the geometric properties of datasets, which can be preserved by geometric transformations. We prove that three types of well-known classification models: kernel methods, SVM classifiers, and linear classifiers, will deliver the same performance over the geometric-transformation-perturbed dataset as over the original dataset. As a result, the geometric perturbation guarantees no loss of accuracy for the discussed classification models. To reach the second goal, we propose a multi-column privacy model to address the particular problems in evaluating privacy quality for multidimensional perturbation. With this privacy model, we analyze three types of inference attacks to geometric perturbation: naive inference, ICA-based reconstruction, and distance-inference attacks. We also develop a randomized optimization algorithm that can progressively find one geometric perturbation that is considerably resilient to the inference attacks. Our experiments show that the geometric perturbation approach can provide high privacy guarantee while maintaining minimum loss of accuracy for the discussed classification models.

In this chapter, we first present the basic challenges in data perturbation techniques for privacy preserving data mining (Section 5.1). Our contributions are focused on geometric perturbation techniques for privacy preserving data perturbation (Section 5.2). The recent developments in data perturbation techniques are briefly reviewed (Section 5.3). In Section 5.4, we describe the properties of geometric rotation transformation and prove that the three most popular categories of classifiers are invariant to rotation transformation. Properties of general linear transformation are also briefly discussed. Section 5.5 introduces a general-purpose privacy measurement model for multi-column

data perturbation and characterizes the privacy property of the rotation-based perturbation in terms of this metric. Three types of inference attacks are analyzed under this privacy model, which results in a randomized optimization algorithm to choose perturbations resilient to the attacks. We present the experimental results to show how the geometric perturbation makes balance between the data quality and the data privacy in Section 5.7.

The next chapter is the application of the geometric data perturbation technique in multiparty collaborative data classification. Most concepts and methods developed in this chapter will also be used in next chapter.

5.1 Data Perturbation and Privacy Preserving Data Mining

We are entering a highly connected information-intensive era. This information age has enabled organizations to collect large amount of data continuously. Many organizations wish to discover and study interesting patterns and trends over the large collections of datasets to improve their productivity and competitiveness. Privacy preserving data mining has become an important enabling technology for integrating data and mining interesting patterns from private collections of databases. This has resulted in a considerable amount of work on privacy preserving data mining methods in recent years [2, 4, 28, 3, 41, 43, 79, 102, 103].

Data perturbation techniques are one of the most popular models for privacy preserving data mining [4, 2]. It is especially useful for applications where the data owners want to participate in cooperative mining but not want to leak the privacy-sensitive information. Typical examples include publishing micro data for research purpose or outsourcing the data to the third party that provides data mining services. A data perturbation procedure can be simply described as follows. Before the data owner publishes the data, they change the data in certain way to disguise the sensitive information while preserving the particular data property that is critical for building meaningful data mining models. Several perturbation techniques have been proposed for mining purpose recently, among which the most popular one is the randomization approach [4, 41]. Different from the randomization approach that focuses on single-dimensional perturbation and assumes independency between data columns, condensation approach [2] and multi-dimensional K-anonymization [75] try to perturb data while preserving the *multi-dimensional information*. In this chapter, we will

propose a new multi-dimensional data perturbation technique specifically for a class of popular data classification mining models.

Loss of Privacy vs. Loss of Information.

Perturbation techniques are often evaluated with two basic metrics, loss of privacy and loss of model-specific information (resulting in loss of accuracy for data classification). An ideal data perturbation algorithm aims at minimizing both privacy loss and information loss. However, the two metrics are not well-balanced in many existing perturbation techniques [4, 3, 40, 2].

Loss of privacy can be intuitively described as the difficulty level in estimating the original values from the perturbed data. The more difficult the original values are estimated, the less loss of privacy is. In [4], the variance of the added random noise is used as the level of difficulty for estimating the original values. However, recent research [40, 3] reveals that variance is not an effective indicator for random noise addition since the original data distribution has to be known $-$ if a particular data distribution is considered, certain part of data in the distribution cannot be effectively protected. In addition, [69] shows that the loss of privacy is also subject to the special attacks that can reconstruct the original data from the perturbed data. k-Anonymization [100] is another popular method for protecting virtual identifiers from the multi-dimensional join of tables. It makes every virtual identifier identify a group of at least k records, thus, hides individual record in the k-record group. By doing so, k-Anonymization limits the effective estimation of the original record into a k-record group assuming that each record in the group is equally protected. However, recent study [81] shows the privacy evaluation for k-Anonymization is far more complicated than the simple assumption.

Loss of information typically refers to the amount of critical information preserved about the datasets after perturbation. However, different data mining tasks, such as classification mining task vs. association rule mining, or different models for the same task, such as decision tree model vs. K-Nearest-Neighbor (KNN) classifier for classification, typically utilize different sets of information about the datasets. For example, the task of building decision trees primarily concerns the column distribution. Hence, the quality of preserving column distribution should become the key in perturbation techniques for decision tree model, as shown in randomization approach [4]. In comparison, the KNN model relies heavily on the distance relationship, which is quite different from the column

distribution. However, existing perturbation techniques do not explicitly address that the critical information is actually task/model-specific. We argue that by narrowing down to preserve only the task/model-specific information, we are able to provide better quality guarantee on both privacy and model accuracy.

We also observed that, many classification models, like KNN model, typically concern the multi-dimensional information rather than single column distribution. Thus, multi-dimensional perturbation techniques with a focus on preserving the model-specific multidimensional information can be more effective for these models. As a result, there is no need (or less complexity) to develop special mining algorithms that can use the perturbed data.

Interesting to note is that the loss of privacy metric and the loss of information metric have exhibited contradictory rather than complimentary results in existing data perturbation techniques [4, 3, 40, 2]. Typically data perturbation algorithms that aim at minimizing the loss of privacy often have to bear with higher information loss. The intrinsic correlation between the loss of privacy and the loss of information raises a number of important issues regarding how to find a right balance between the two measures and how to build a data perturbation algorithm that ensures desired privacy requirements and yet minimizes the loss of information for the specific data mining task.

5.2 The Scope and Contributions of This Chapter

Bearing these issues in mind, we have developed a random geometric perturbation approach to privacy preserving data classification. In contrast to other existing privacy-preserving classification methods [2, 4, 43, 79], our random geometric-transformation based perturbation exploits the task-specific multi-dimensional information about the datasets to be classified, which is critical to a large category of classification algorithms, and aims at producing a robust data perturbation that exhibits a better balance between loss of privacy and loss of information.

Concretely, we observe that the multi-dimensional geometric properties of datasets are the critical "task-specific information" for many classification algorithms. By preserving multi-dimensional geometric properties of the original dataset, classifiers trained over the perturbed dataset presents

the same quality as classifiers over the original dataset. One intuitive way to preserve the multi-dimensional geometric properties is to perturb the original dataset through random geometric transformation such as random rotation and translation. We have identified and proved that kernel methods, SVM classifiers with the three popular kernels, and linear classifiers, are the three categories of classifiers that are "rotation/translation-invariant".

Another important challenge for the random geometric perturbation approach is the privacy loss measurement (the level of uncertainty) and privacy assurance (the resilience of the geometric transformation against unauthorized disclosure). Given that a random geometric-transformation based perturbation is a multi-dimensional perturbation, the privacy guarantee of the multiple dimensions (attributes) should be evaluated collectively to ensure the privacy of all columns involved and the privacy of the multi-column correlations. We design a unified privacy model to tackle the problem of privacy evaluation for multi-dimensional perturbation, which addresses three types of possible attacks: direct estimation, approximate reconstruction, and distribution-based inference attacks.

With the unified privacy metric, we present the privacy assurance of the random geometric perturbation as an optimization problem: given that all rotation/translation transformations result in zero-loss of accuracy for the discussed classifiers, we want to pick one pair of rotation/translation matrices that provide higher privacy guarantee and stronger resilience against the three types of inference attacks than most other pairs. Our experiments demonstrate that, with the attack-resilient perturbation selection algorithm, the resultant perturbation can achieve considerably higher privacy guarantee and be more robust in countering inference attacks than other existing multi-dimensional perturbation techniques.

5.3 Related Work

A considerable amount of work on privacy preserving data mining methods have been reported in recent years [2, 4, 28, 3, 41, 103]. The most relevant work about perturbation techniques for data mining includes the random noise addition methods [4, 41] and the condensation-based perturbation [2], while K-Anonymization [100] is another perturbation technique for general-purpose data sharing that can also be potentially used for some specific mining models [46]. Since our approach is more related to the first two perturbations, we below focus our discussion on them and discuss

their weakness in the context of privacy preserving data classification.

Random Noise Addition Approach Most of the existing randomization techniques for data perturbation are value-based randomization. This type of techniques relies on the facts that 1) Data owners may not want to equally protect all values in a record, thus a column-based value distortion can be applied to perturb some sensitive columns. 2) Data classification models do not necessarily require the individual records, but only the column value distributions [4], assuming that the columns are independent. The basic goal is to disguise the original values by injecting certain amount of random noise, while the specific information, such as the column distribution, can still be effectively estimated from the perturbed data for data mining purposes.

A typical random noise addition model [4] can be precisely described as follows. We treat the original values $(x_1, x_2, \ldots, x_n)$ from a column to be randomly drawn from a random variable $\mathbf{X}$, which has some kind of distribution. The randomization process changes the original data by adding random noises $\mathbf{R}$ to the original data values, and generates a perturbed data column $\mathbf{Y}$, $\mathbf{Y} = \mathbf{X}+\mathbf{R}$. The resulting tuples $(x_1 + r_1, x_2 + r_2, \ldots, x_n + r_n)$ and the distribution of $\mathbf{R}$ are published. The key of random noise addition is the distribution reconstruction algorithm [4, 3], which is able to construct the column distribution of $\mathbf{X}$ based on the perturbed data $\mathbf{R}$ and the distribution of $\mathbf{R}$, so that data mining models that rely on the column distributions can still be developed with the perturbed data. The randomization approach is also generalized by [40] and [5]. [40] proposes a refined privacy metric for the general randomization approach, and [5] develops a framework based on the refined privacy metric to improve the balance between the privacy and accuracy.

Special mining algorithms, such as a decision-tree algorithm [4], and an association-rule mining algorithm [41], which meet the assumption of independent columns and are able to reconstruct the column distributions from perturbed datasets, are developed for the randomization approach.

While the randomization approach is intuitive, several researchers have recently identified privacy breaches as one of the major problems with the randomization approach. Kargupta et al. [69, 61] observed that the spectral properties of the randomized data can be utilized to separate noise from the private data. A filtering algorithm based on random matrix theory is used to approximately reconstruct the private data from the perturbed data. The authors demonstrated that the randomization approach preserves little privacy in many cases.

Furthermore, there has been research [2] addressing other weaknesses associated with the value based randomization approach. First, most of existing randomization and distribution reconstruction algorithms only concern about preserving the distribution of single column. There has been surprisingly little attention paid on preserving value distributions over multiple correlated dimensions. Second, the randomization approach also requires us to develop new distribution-based classification algorithms, which is very inconvenient in practice.

Our random geometric perturbation approach takes advantage of the protection provided by randomization, but it can be directly applied to three categories of popular classification models without any change or redevelopment of classification algorithms. Most importantly, since the assumption of independent columns is usually not held for real datasets, other classification models that utilize the multidimensional information can potentially benefit from our perturbation technique.

Condensation-based perturbation approach The condensation approach [2] is a typical multidimensional perturbation technique, which aims at preserving the covariance matrix for multiple columns. Thus, some geometric properties such as the shape of decision boundary are approximately preserved. Different from the randomization approach, it perturbs multiple columns as a whole to generate entire "perturbed dataset". The authors argue that the perturbed dataset preserves the covariance matrix, and thus, most existing data mining algorithms can be applied directly to the perturbed dataset without requiring any change or new development of algorithms, which is also a benefit shared by our geometric perturbation.

The condensation approach can be briefly described as follows. It starts by partitioning the original data into k-record groups. Each group is formed by two steps – randomly selecting a record from the existing records as the center of group, and then finding the $(k-1)$ nearest neighbors of the center to be the other $(k-1)$ members. The selected k records are removed from the original dataset before forming the next group. Since each group has small locality, it is possible to regenerate a set of k records to approximately preserve the distribution and covariance. The record regeneration algorithm tries to preserve the eigenvectors and eigenvalues of each group. As a result, the distribution and the covariance of the points in the group are approximately preserved as shown in Figure 74. The authors demonstrated that the condensation approach can preserve data covariance well, and thus will not significantly sacrifice the accuracy of classifiers if the classifiers are trained with

the perturbed data.

However, we have observed that the condensation approach is weak in protecting the private data. The KNN-based data groups result in some serious conflicts between preserving covariance information and preserving privacy. As stated by the authors, the smaller the size of the locality is in each group, the better the quality of preserving the covariance with the regenerated k records is. Note that the regenerated k records are confined in the small spatial locality as shown in Figure 74. It follows that the smaller the locality is, the closer the regenerated records are to the original records. We design an algorithm that tries to find the nearest neighbor in the original data for each regenerated record. The result (section 5.7) shows that the difference between the regenerated records and the nearest neighbor in original data are very small, and thus, the original data records can be estimated from the perturbed data with high confidence.

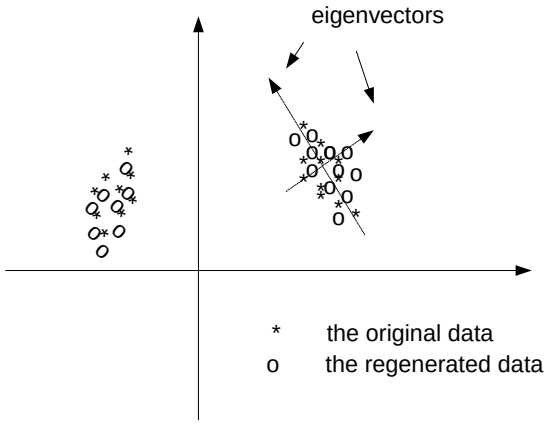

Figure 74: Condensation approach

Condensation approach in fact tries to preserve the geometric decision boundary, however, in a more conservative way, which greatly limits the privacy guarantee it can provide as shown in experiments. By comparison, our geometric perturbation provides higher privacy guarantee than the condensation approach, while precisely preserving the model accuracy as well.

5.4 Geometric Transformations and Data Classification

In this section, we first identify the geometric properties of the datasets that are significant to most classification algorithms. Then we describe the definition of geometric perturbation, and discuss

the effect of geometric transformations to three categories of popular classification models. In particular, we focus on geometric rotation and translation. The discussion will be extended to general non-singular linear transformations as well. Apparently, geometric perturbation works only for *numerical* data classification. Therefore, by default, the datasets discussed in this research are all numerical data. Before entering concrete discussion, we define the notations for the datasets used in data classification.

Training Dataset and Unclassified Dataset. Training dataset is the part of data that has to be exported/published in privacy-preserving data classification. A classifier learns the classification model from the training data and then is applied to classify the unclassified data. Suppose that X is a training dataset consisting of N data rows (records) and d columns (attributes, or dimensions). For the convenience of mathematical manipulation, we use $X_{d\times N}$ to denote the dataset, i.e., $X = [\mathbf{x}_1 \dots \mathbf{x}_N]$, where $\mathbf{x}_i$ is a data tuple, representing a vector in the real space $\mathbb{R}^d$. Each data tuple $\mathbf{x}_i$ belongs to a predefined class, which is indicated by the class label attribute y_i. The class label can be nominal (or continuous for regression), which is public, i.e., privacy-insensitive. All other attributes containing private information needs to be protected. Unclassified dataset could also be exported/published with privacy-protection if necessary.

We also consider X is a sample dataset from the d-dimension random vector $\mathbf{X} = [\mathbf{X}_1, \mathbf{X}_2, \dots, \mathbf{X}_d]^T$. As a convention, we use bold $\mathbf{X}$ to represent the corresponding random vector of the dataset X. To unify the notations, we use bold lower case to represent vectors, bold upper case to represent random variables, and upper case to represent matrices.

Geometric Transformations Let $R_{d\times d}$ represent a *rotation matrix*, which will be described in detail later. Let $\mathbf{t}_{d\times 1}$ represent a translation vector. we also define the *translation matrix* as follows.

Definition 1. *Ψ is a translation matrix if $\Psi = [\mathbf{t}, \mathbf{t}, \dots, \mathbf{t}]_{d\times n}$, i.e., $\Psi_{d\times n} = \mathbf{t}_{d\times 1}\mathbf{1}^t_{N\times 1}$.*

where $\mathbf{1}_{N\times 1}$ is the vector of N '1's. Let $\Delta_{d\times N}$ be a random noise matrix, where each element is Independently and Identically Distributed (iid) Gaussian variable $\delta_{ij} \sim N(0, \sigma^2)$. A general form of the geometric perturbation of the dataset X is denoted as a function $G(X)$, $G(X) = RX + \Psi + \Delta$. Note that the transformation will not change the class label of a data record, i.e., the transformation of data record $\mathbf{x}_i$, still has the label y_i.

We will first discuss the properties of the basic-form geometric perturbation, i.e., $G(X) = RX + \Psi$, while the noise component Δ will be added later in enhancing the privacy guarantee of geometric perturbation. The following discussion will be focused on the two components, concerning rotation perturbation and translation perturbation, respectively.

5.4.1 Properties of Geometric Rotation

A rotation matrix $R_{d\times d}$ is a matrix with following properties. Let R^T represent the transpose of the matrix R, r_{ij} represent the (i, j) element of R, and I be the identity matrix. The rows and columns of R are *orthonormal* [90], i.e., for any column j, $\sum_{i=1}^{d} r_{ij}^2 = 1$, and for any two columns j and k, $j \neq k$, $\sum_{i=1}^{d} r_{ij}r_{ik} = 0$. A similar property is held for rows. The definition infers that $R^T R = RR^T = I$. It also implies that by changing the order of the rows or columns of a rotation matrix, the resulting matrix is still a rotation matrix. A random rotation matrix can be efficiently generated following the Haar distribution [99].

A key feature of rotation transformation is preserving Euclidean distance. Let $\mathbf{x}^T$ represent the transpose of vector $\mathbf{x}$, and $\| \mathbf{x} \| = \mathbf{x}^T\mathbf{x}$ represent the length of a vector $\mathbf{x}$. By the definition of rotation matrix, we have $\| R\mathbf{x} \| = \| \mathbf{x} \|$ Similarly, inner product is also invariant to rotation. Let $< \mathbf{x}, \mathbf{y} > = \mathbf{x}^T\mathbf{y}$ represent the inner product of $\mathbf{x}$ and $\mathbf{y}$. We have $< R\mathbf{x}, R\mathbf{y} >$ = $\mathbf{x}^T R^T R\mathbf{y} =< \mathbf{x}, \mathbf{y} >$.

In general, rotation also preserves the geometric shapes such as hyperplane and hyper curved surface in the multidimensional space. We observed that since many classifiers look for geometric decision boundary, such as hyperplane and hyper surface, rotation transformation will preserve the most critical information for many classification models. We will discuss several such classification models later.

5.4.2 Rotation-invariant Classifiers

We first define the concept of "transformation-invariant classifiers", and then discuss the concrete classifiers having this property. We say a classification algorithm is invariant to a transformation, if the classifier trained with the transformed data has a similar accuracy rate as that trained with the original data. We formally define the transformation-invariant classifier as follows.

A classification problem is also a function approximation problem – classifiers are the functions

learned from the training data [57]. In the following discussion, we use functions to represent classifiers. Let $\hat{f}_X$ represent a classifier $\hat{f}$ trained with dataset X and $\hat{f}_X(Y)$ be the classification result on dataset Y. Let $T(X)$ be any transformation function, which transforms the dataset X to another dataset X_T. We use $Err(\hat{f}_X(Y))$ to denote the error rate of classifier $\hat{f}_X$ on testing data Y and let ε be some small real number, $|\varepsilon| < 1$.

Definition 2. *A classifier $\hat{f}$ is invariant to a transformation T if and only if $Err(\hat{f}_X(Y)) = Err(\hat{f}_{T(X)}(T(Y))) + \varepsilon$ for any training dataset X and testing dataset Y.*

With the strict condition $\hat{f}_X(Y) \equiv \hat{f}_{T(X)}(T(Y))$, we get the corollary 8.

Corollary 8. *In particular, if $\hat{f}_X(Y) \equiv \hat{f}_{T(X)}(T(Y))$ is satisfied for any training dataset X and testing dataset Y, the classifier is invariant to the transformation $T(X)$.*

Specifically, if a classifier $\hat{f}$ is invariant to *rotation* transformation, we name it as a *rotation-invariant classifier*, similar definition for a *translation-invariant classifier*.

In subsequent sections, we will prove that kernel methods, SVM classifiers with certain kernels, and hyperplane-based classifiers, are three types of rotation-invariant classifiers, based on the strict condition given by Corollary 8.

5.4.2.1 KNN Classifiers and Kernel Methods

A K-Nearest-Neighbor (KNN) classifier determines the class label of a point by looking at the labels of its k nearest neighbors in the training dataset and classifies the point to the class that most of its neighbors belong to. Since the distances between any points are not changed with rotation transformation, the k nearest neighbors are not changed and thus the classification result is not changed either. We have the first conclusion about the KNN classifiers.

Lemma 9. *KNN classifiers are rotation-invariant.*

KNN classifier is a special case of kernel methods. We assert that any kernel methods will be invariant to rotation, too. Same as the KNN classifier, a typical kernel method [1] is a local classification method, which classifies the new data record only based on the information of its neighbors in the training data.

[1] SVM is also a kind of kernel method, but its training process is different from the kernel methods we discuss here.

Theorem 10. *Kernel methods are invariant to rotation.*

`Sketch of Proof.` Let us formally define kernel methods first. Like KNN classifiers, a kernel method also estimates the class label of a point $\mathbf{x}$ with the class labels of its neighbors. Let $K_\lambda(\mathbf{x}, \mathbf{x}_i)$ be the kernel function weighting any point $\mathbf{x}_i$ in $\mathbf{x}$'s neighborhood and λ define the geometric width of the neighborhood. We assume $\{\mathbf{x}_1, \mathbf{x}_2, \ldots, \mathbf{x}_n\}$ be the points in the $\mathbf{x}$'s neighborhood determined by λ. A kernel classifier for continuous class labels[2] is defined as

$$\hat{f}_X(\mathbf{x}) = \frac{\sum_{i=1}^{n} K_\lambda(\mathbf{x}, \mathbf{x}_i) y_i}{\sum_{i=1}^{n} K_\lambda(\mathbf{x}, \mathbf{x}_i)} \tag{20}$$

Specifically, the kernel $K_\lambda(\mathbf{x}, \mathbf{x}_i)$ is defined as

$$K_\lambda(\mathbf{x}, \mathbf{x}_i) = D(\frac{\parallel \mathbf{x} - \mathbf{x}_i \parallel}{\lambda}) \tag{21}$$

$D(t)$ is a function, for example, the Gaussian kernel $D(t) = \frac{1}{\sqrt{2\pi}} \exp\{-t^2/2\}$. Since $\parallel R\mathbf{x} - R\mathbf{x}_i \parallel = \parallel \mathbf{x} - \mathbf{x}_i \parallel$ and λ is constant, $D(t)$ is not changed after rotation transformation and, thus, $K_\lambda(R\mathbf{x}, R\mathbf{x}_i) = K_\lambda(\mathbf{x}, \mathbf{x}_i)$. Since the geometric area around the point is also not changed, the point set in the neighborhood of $R\mathbf{x}$ are still the rotation of those in the neighborhood of $\mathbf{x}$, i.e. $\{\mathbf{x}_1, \mathbf{x}_2, \ldots, \mathbf{x}_n\} \Rightarrow \{R\mathbf{x}_1, R\mathbf{x}_2, \ldots, R\mathbf{x}_n\}$ and these n points are used in $\hat{f}_{RX}$, which makes $\hat{f}_{RX}(R\mathbf{x}) = \hat{f}_X(\mathbf{x})$. □

5.4.2.2 Support Vector Machines

Support Vector Machine (SVM) classifiers also utilize kernel functions in training and classification. However, it has an explicit training procedure, which generates a classifier for classification. Let y_i be the class label to a tuple $\mathbf{x}_i$ in the training set, α_i and β_0 be the parameters determined by training. A SVM classifier calculates the classification result of $\mathbf{x}$ using the following function.

$$\hat{f}_X(\mathbf{x}) = \sum_{i=1}^{N} \alpha_i y_i K(\mathbf{x}, \mathbf{x}_i) + \beta_0 \tag{22}$$

Different from kernel methods, which do not have a explicit training procedure, we shall prove that SVM classifiers are invariant to rotation in two steps, 1) training with the rotated dataset generates the same set of parameters α_i and β_0; 2) the classification function $\hat{f}$ is invariant to rotation.

[2] It has different form for discrete class labels, but the proof will be similar.

Theorem 11. *SVM classifiers using polynomial, radial basis, and sigmoid kernels are invariant to rotation.*

Proof. The SVM training problem is an optimization problem, which finds the parameters α_i and β_0 to maximize the Lagrangian (Wolfe) dual objective function [57]

$$L_D = \sum_{i=1}^{N} \alpha_i - 1/2 \sum_{i,j=1}^{N} \alpha_i \alpha_j y_i y_j K(\mathbf{x}_i, \mathbf{x}_j),$$

subject to:

$$0 < \alpha_i < \gamma, \qquad \sum_{i=1}^{N} \alpha_i y_i = 0,$$

where γ is a parameter chosen by the user to control the allowed errors around the decision boundary. We see that the training result of α_i is only determined by the form of kernel function $K(\mathbf{x}_i, \mathbf{x}_j)$. With the determined α_i, β_0 can be determined by solving $y_i \hat{f}_X(\mathbf{x}_i) = 1$ for any $\mathbf{x}_i$ [57], which is again determined by the kernel function. It is clear that if $K(R\mathbf{x}, R\mathbf{x}_i) = K(\mathbf{x}, \mathbf{x}_i)$ is held, the training procedure generates the same set of parameters.

There are the three popular choices for kernels discussed in the SVM literature [33, 57].

$$\begin{aligned} \text{d-th degree polynomial:} \quad & K(\mathbf{x}, \mathbf{x}') = (1 + < \mathbf{x}, \mathbf{x}' >)^d, \\ \text{radial basis:} \quad & K(\mathbf{x}, \mathbf{x}') = \exp(- \parallel \mathbf{x} - \mathbf{x}' \parallel / c), \\ \text{sigmoid:} \quad & K(\mathbf{x}, \mathbf{x}') = \tanh(\kappa_1 < \mathbf{x}, \mathbf{x}' > + \kappa_2) \end{aligned}$$

Note that the three kernels only involve distance and inner product calculation. As we discussed in section 5.4.1, the two operations keep invariant to the rotation transformation. Thus, $K(R\mathbf{x}, R\mathbf{x}') = K(\mathbf{x}, \mathbf{x}')$ is held for the three kernels, and, thus, training with the rotated data will not change the parameters for the SVM classifiers using the three popular kernels.

It is easy to verify $\hat{f}_X(\mathbf{x}) = \hat{f}_{RX}(R\mathbf{x})$ for the classification function (22), which involves only the invariant parameters and the invariant kernel functions. □

5.4.2.3 Perceptron

Perceptron is a typical linear classification model. We use perceptron as a representative example for all hyperplane-based linear classifiers. The following discussion for perceptron can be safely extended to general hyperplane-based linear classifiers.

A perceptron classifier uses a hyperplane to separate the training data, with the weights $\mathbf{w}^T = [w_1, \ldots, w_d]$ and a bias β_0. The weight and bias parameters are determined by the training procedure. A trained classifier is represented as follows.

$$\hat{f}_X(\mathbf{x}) = \mathbf{w}^T\mathbf{x} + \beta_0$$

Theorem 12. *Perceptron classifiers are invariant to rotation.*

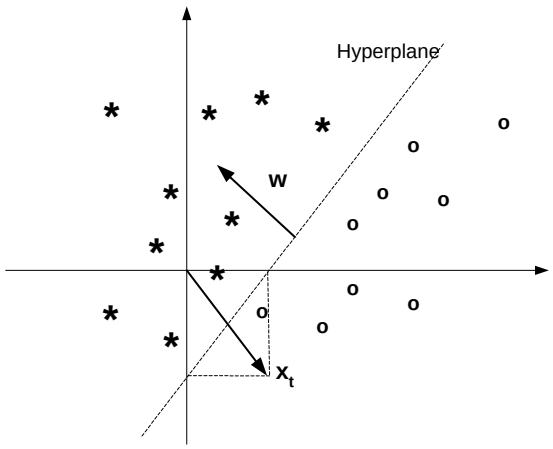

Figure 75: Hyperplane and its parameters

Proof. It is important to understand the relationship between the parameters and the hyperplane. As Figure 75 shows, the hyperplane can be represented as $\mathbf{w}^T(\mathbf{x} - \mathbf{x_t}) = 0$, where $\mathbf{w}$ is the perpendicular axis to the hyperplane, and $\mathbf{x}_t$ represents the deviation of the plane from the origin (i.e., $\beta_0 = -\mathbf{w}^T\mathbf{x}_t$).

Intuitively, rotation will rotate the classification hyperplane and feature vectors. The perpendicular axis $\mathbf{w}$ is changed to $R\mathbf{w}$ and the deviation $\mathbf{x}_t$ becomes $R\mathbf{x}_t$ after rotation. Let $\mathbf{x}^r$ represent the data in the rotated space. Then, the rotated hyperplane is represented as $(R\mathbf{w})^T(\mathbf{x}^r - R\mathbf{x}_t) = 0$, and the classifier is transformed to $\hat{f}_{RX}(\mathbf{x}^r) = \mathbf{w}^T R^T(\mathbf{x}^r - R\mathbf{x}_t)$. Since $\mathbf{x^r} = R\mathbf{x}$ and $R^T R = I$, $\hat{f}_{RX}(\mathbf{x}^r) = \mathbf{w}^T R^T R(\mathbf{x} - \mathbf{x_t}) = \mathbf{w}^T(\mathbf{x} - \mathbf{x_t}) = \hat{f}_X(\mathbf{x})$. The two classifiers are equivalent. □

In general, since rotation transformation will preserve distance, density, and geometric shapes, any classifiers that find the decision boundary based on the geometric properties of the dataset, will still find the rotated geometric decision boundary. In fact, the distance relationship is critical in KNN classifiers and the kernel methods to define the decision area, hyperplanes are the geometric decision

boundaries for linear classifiers, and different kernels in SVM classifiers use either hyperplanes or hyper curved surfaces as the decision boundary [33]. Therefore, these classifiers are all rotation-invariant classifiers.

5.4.3 Random Translation Perturbation

From above discussion, it is easy to understand and to prove that those classifiers that depend on distance relationship and hyperplanes are also invariant to translation. However, some classifiers, such as, SVM classifiers with polynomial kernels and sigmoid kernels will be slightly affected by random translation. We will show this in experiments.

Random translation Ψ is also a necessary component of basic geometric perturbation, $G(X) = RX + \Psi$. Without random translation, the rotation center is not protected - the points around the origin will be perturbed less and thus the perturbation preserves less privacy for such points. As we will see in section 5.5.4, translation perturbation is used to effectively protect the rotation center from the rotation-center-oriented attacks.

It is easy to understand that classifiers that depend on distance relationship and hyperplanes are also invariant to translation. For those geometric classifiers depending on inner product, such as SVM classifiers with polynomial kernels and sigmoid kernels, the strict condition $\hat{f}_X(Y) \equiv \hat{f}_{T(X)}(T(Y))$ cannot be satisfied for translation transformation, because affine transformation (see Section 2.1) does not preserve inner product. However, based on our discussion on the common features of geometric classifiers, we conjecture that these geometric classifiers will also be invariant to translation. The experimental result supports this hypothesis.

5.5 Evaluating Privacy Quality for Random Geometric Perturbation

The goals of random geometric perturbation are twofold: preserving the accuracy of classifiers, and preserving the privacy of data. The discussion about the rotation-invariant classifiers has proven that rotation transformation theoretically guarantees zero-loss of accuracy for three popular types of classifiers. It is similar to translation transformation. As a result, numerous geometric perturbations can present the same model accuracy, and we only need to find one that maximizes the privacy guarantee in terms of various potential attacks.

We dedicate this section to discuss how good the geometric perturbation approach is in terms of preserving privacy. Since single translation is too simple to protect the original data, we will primarily consider rotation transformation or its combination with translation. The critical step is to define a multi-column privacy measure for evaluating the privacy quality of a rotation perturbation to a given dataset. With this privacy measure, we can employ certain optimization methods to find a good one among an infinite number of rotation perturbations for a given dataset. This multi-column privacy model can also be conveniently applied to evaluate the efficiency of attacks to multi-dimensional perturbation techniques.

5.5.1 Multidimensional Privacy Evaluation Model

Unlike existing randomization methods, where multiple columns are perturbed separately, the random geometric perturbation needs to perturb *all* columns together. Therefore, the privacy quality of all columns is correlated under one single transformation and should be evaluated under a unified metric. Our approach to evaluating the privacy quality of random rotation perturbation consists of two steps: first, we define a unified general-purpose privacy metric that is effective for any multi-dimensional perturbation technique. Second, we present the methodology of privacy evaluation with the unified privacy metric in terms of inference attacks.

Conceptual Multidimensional Privacy Evaluation Model Since in practice different columns(attributes) may have different privacy concern, we consider that the general-purpose privacy metric Φ for entire dataset is based on **column privacy metric**. An abstract privacy model is defined as follows. Let $\mathbf{p}$ be the column privacy metric vector $\mathbf{p} = [p_1, p_2, \ldots, p_d]$, and there are **privacy weights** associated to the d columns, respectively, denoted as $\mathbf{w} = (w_1, w_2, \ldots, w_d)$. $\Phi = \Phi(\mathbf{p}, \mathbf{w})$ uses the two vectors to define the privacy guarantee. In summary, the design of specific privacy model should determine the three factors $\mathbf{p}$, $\mathbf{w}$, and the function Φ.

We will leave the discussion about one concrete design of $\mathbf{p}$ later, and define the other two factors first. The first design idea is to take the column importance into unification of different column privacy. Intuitively, the more important the column is, the higher level of privacy guarantee will be required for the perturbed data column. Since $\mathbf{w}$ is used to denote the importance of columns in terms of preserving privacy, we use p_i/w_i to represent the *weighted column privacy* of column i.

The second intuition is the concept of *minimum privacy guarantee* and *average privacy guarantee* among all columns. Normally, when we measure the privacy quality of a multidimensional perturbation, we need to pay more attention to the column that has the lowest weighted column privacy, because such a column could become the breaking point of privacy. Hence, we design the first composition function $\Phi_1 = \min_{i=1}^{d}\{p_i/w_i\}$ and call it the minimum privacy guarantee. Similarly, the *average privacy guarantee* of the multi-column perturbation, defined by $\Phi_2 = \frac{1}{d}\sum_{i=1}^{d} p_i/w_i$, could be another interesting measure.

Variance-based Unified Column Privacy Metric Intuitively, for a data perturbation approach, the quality of preserved privacy can be understood as the difficulty level of estimating the original data from the perturbed data. Therefore, how different the *estimated data* is from the original data could be an intuitive measure. We use a variance-of-difference (VoD) based approach, which is derived from the naive variance-based evaluation [4] with more general setting.

Let the difference between the original column data and the estimated data be a random variable $\mathbf{D}_i$. Without any knowledge about the original data, the mean and variance of the difference present the quality of the estimation. Since the mean of difference can be easily removed if the attacker can estimate the original distribution of column, we use only the variance of the difference (VoD) as the primary metric to determine the level of difficulty in estimating the original data.

VoD is formally defined as follows. Let $\mathbf{X}_i$ be a random variable representing the column i, $\mathbf{X}_i'$ be the estimated result of $\mathbf{X}_i$, and $\mathbf{D}_i$ be $\mathbf{D}_i = \mathbf{X}' - \mathbf{X}$. Let $E[\mathbf{D}_i]$ and $Var(\mathbf{D}_i)$ denote the mean and the variance of $\mathbf{D}$ respectively. Then VoD for column i is $Var(\mathbf{D}_i)$. Let an estimation of certain value, say x_i, be x_i' in $\mathbf{X}_i'$, $\sigma = \sqrt{Var(\mathbf{D}_i)}$, and c denote confidence parameter depending on the distribution of $\mathbf{D}_i$ and the corresponding confidence level. The corresponding original value x_i in $\mathbf{X}_i$ is located in the range defined below:

$$[x_i' - E[\mathbf{D}_i] - c\sigma, x_i' - E[\mathbf{D}] + c\sigma]$$

Without considering $E[\mathbf{D}]$, the width of the estimation range, $2c\sigma$, presents the difficulty of guessing the original value, which proportionally reflects the level of privacy guarantee. For simplicity, we often use only VoD or σ to represent the privacy level.

VoD only defines the privacy guarantee for single column. As we have discussed, we need

to evaluate the privacy of all perturbed columns together. The single-column VoD does not work across different columns since different column value ranges may result in very different VoDs. Therefore, the same amount of VoD is not equally effective for columns with different value ranges. One straightforward method to unify the different value ranges is via *normalization* over the original dataset and the perturbed dataset. Normalization can be done with max/min normalization or standardized normalization [83]. We use max/min normalization in this paper.

Attack Evaluation: Since the variance-based model evaluates the accuracy of "estimated" values. It is convenient to incorporate attack evaluation into privacy evaluation. In general, let X be the normalized original dataset, P be the perturbed dataset, and O be the estimated/observed dataset. We calculate $VoD(\mathbf{X}_i, \mathbf{O}_i)$ for the column i in terms of different attacks. Here, we summarize the evaluation of the inference attacks to rotation perturbation [25] that will be described in the later sections.

1. Naive Estimation: $O = P$;

2. ICA-based Reconstruction: Independent Component Analysis (ICA) is used to estimate R. Let $\hat{R}$ be the estimate of R, and align the estimated data $\hat{R}^{-1}P$ with the known column distributions and statistics to get the dataset O;

3. Distance-based Inference: knowing a set of special points in X that can be mapped to certain set of points in P, so that the mapping helps to get the estimated rotation $\hat{R}$, and then $O = \hat{R}^{-1}P$.

5.5.2 Naive Estimation Attack

With the VoD metric over the normalized data, we can formally analyze the privacy guarantee provided by the rotation perturbed data, if no additional information is known by the attacker. Let X be the normalized dataset, X' be the rotation of X, and I_d be the d-dimensional identity matrix. Thus, VoD of column i can be evaluated by

$$\begin{aligned} Cov(\mathbf{X}' - \mathbf{X})_{(i,i)} &= Cov(R\mathbf{X} - \mathbf{X})_{(i,i)} \\ &= ((R - I_d)Cov(\mathbf{X})(R - I_d)^T)_{(i,i)} \end{aligned} \quad (23)$$

Let r_{ij} represent the element (i, j) in the matrix R, and c_{ij} be the element (i, j) in the covariance matrix of $\mathbf{X}$. The VoD for ith column is computed as follows.

$$Cov(\mathbf{X}' - \mathbf{X})_{(i,i)} = \sum_{j=1}^{d}\sum_{k=1}^{d} r_{ij}r_{ik}c_{kj} - 2\sum_{j=1}^{d} r_{ij}c_{ij} + c_{ii} \tag{24}$$

When the random rotation matrix is generated following the Haar distribution, a considerable number of matrix entries are approximately independent normal distribution $N(0, 1/d)$ [67]. The full discussion about the numerical characteristics of the random rotation matrix will be out of the scope of this work. For simplicity and easy understanding, we assume that all entries in random rotation matrix approximately follow independent normal distribution $N(0, 1/d)$. Therefore, random rotations will make VoD_i changing around the mean value c_{ii} as shown in the following equation.

$$E[VoD_i] \sim \sum_{j=1}^{d}\sum_{k=1}^{d} E[r_{ij}]E[r_{ik}]c_{kj} - 2\sum_{j=1}^{d} E[r_{ij}]c_{ij} + c_{ii} = c_{ii}$$

It means that the original column variance could substantially influence the result of random rotation. However, the expectation of VoDs is not the only factor determining the final privacy guarantee. We should also look at the variance of VoDs. If the variance of VoDs is considerably large, we still get great chance to find a rotation with high VoDs in a set of sample random rotations, and the larger the $Var(VoD_i)$ is, the more likely the randomly generated rotation matrices can provide a high privacy level.

$$\begin{aligned} Var(VoD_i) &\sim \sum_{i=1}^{d}\sum_{j=1}^{d} Var(r_{ij})Var(r_{ik})c_{ij}^2 \\ &\quad +4\sum_{j=1}^{d} Var(r_{ij})c_{ij}^2 \\ &\sim O(1/d^2\sum_{i=1}^{d}\sum_{j=1}^{d} c_{ij}^2 + 4/d\sum_{j=1}^{d} c_{ij}^2). \end{aligned}$$

The above result shows that $Var(VoD_i)$ is approximately related to the average of the squared covariance entries, with more influence from the row i of covariance matrix. Therefore, by looking at the covariance matrix of the original dataset, we see the chance to find a random rotation that can give high privacy guarantee.

In Equation 24, we also notice that the i-th row vector of rotation matrix, i.e., the values r_{i*}, plays a dominating role in calculating VoD_i. Since swapping rows of a rotation matrix will result in another rotation matrix, it is possible to simply swap the rows of R to locally improve the

privacy guarantee. This drives us to propose the row-swapping based local optimization method for finding a better rotation from a given rotation matrix. We define the method as follows. Let $\{(1), (2), \ldots, (d)\}$ be a permutation of the sequence $\{1, 2, \ldots, d\}$. The goal is to find a permutation of rows that maximizes the minimum (or average) privacy guarantee.

$$argmax_{\{(1),(2),\ldots,(d)\}}\{min_{1\leq i\leq d}\{(\sum_{j=1}^{d}\sum_{k=1}^{d} r_{(i)j}r_{(i)k}c_{kj} - 2\sum_{j=1}^{d} r_{(i)j}c_{ij} + c_{ii})/w_i\}\} \quad (25)$$

5.5.3 ICA-based Attack

Naive estimation is the basic attack trying to find the original value directly from the perturbed data, which will be ineffective to carefully perturbed data. In this section, we introduce a high-level attack based on data reconstruction. The basic method trying to reconstruct X from the perturbed data RX would be Independent Component Analysis (ICA) technique derived from signal processing [63].

The ICA model can be applied to estimate the independent components (the row vectors) of the original dataset X, from the perturbed data, if the following conditions are satisfied:

1. The source row vectors are independent;
2. All source row vectors should be non-Gaussian with possible exception of one row;
3. The number of observed row vectors must be at least as large as the independent source row vectors.
4. The transformation matrix R must be of full column rank.

For rotation matrices, the 3rd and 4th conditions are always satisfied. However, the first two conditions although practical for signal processing, are often not satisfied in data classification. Furthermore, there are a few more difficulties in applying the above ICA-based attack. First of all, even ICA can be done successfully, the order of the original independent components cannot be preserved or determined through ICA [63]. Formally, any permutation matrix P and its inverse P^{-1} can be substituted in the model to give $X' = RP^{-1}PX$. ICA could possibly give the estimate for some permutated source PX. Thus, we cannot identify the particular column if the original

column distributions are unknown. Second, even if the ordering of columns can be identified, ICA reconstruction does not guarantee to preserve the variance of the original signal – the estimated signal is often scaled up but we do not know how much the scaling is unless we know the original value range of the column. Therefore, without knowing the basic statistics of original columns, ICA-attack is not effective.

However, as we have mentioned earlier, such column statistics are not impossible to get in similar datasets provided for privacy-preserving data mining. We assume the attackers know the basic statistics, including the max/min values and the PDF of each column. The enhanced ICA-based attack can be described as follows.

1. Run ICA algorithm to get a reconstructed dataset;
2. For each reconstructed column $\mathbf{O}_i$ and each original column $\mathbf{X}_j$, we scale $\mathbf{O}_i$ with the max/min values of $\mathbf{X}_j$, and compare the PDFs of the scaled $\mathbf{O}_i$ and $\mathbf{X}_j$ to find the closest match;

The important step is "PDF Alignment" to find the match between original columns and the perturbed columns. A direct method is to calculate the difference between the two PDF functions. Let $f(x)$ and $g(x)$ be the original PDF and the PDF of the reconstructed column, respectively. A typical method to define the difference of PDFs employs the following function.

$$\Delta PDF = \int |f(x) - g(x)| dx \tag{26}$$

In practice, for easy manipulation we discretize the PDF function into bins. It is then equivalent to use the discretized version: $\sum_{i=1}^{n} |f(b_i) - g(b_i)|$, where b_i is the discretized bin i. However, the evaluation is not accurate if the values in the two columns are not in the same range as shown in Figure 76. Hence, the reconstructed PDF needs to be translated and scaled to match the range, which requires the maximum/minimum values of the original column to be known, too.

The above procedure describes how to use ICA and additional knowledge about the original dataset to precisely reconstruct the original dataset. Note if the four conditions for effective ICA are exactly satisfied and the basic statistics and PDFs are all known, the basic rotation perturbation will be broken by the enhanced ICA-based attack. In practice, we can test if the first two conditions

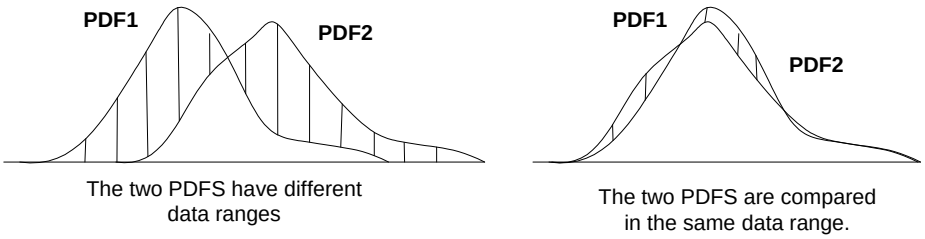

Figure 76: Comparing PDFs in different ranges results in large error. (The lined areas are calculated as the difference between the PDFs.)

for effective ICA are satisfied to decide whether we can safely use rotation perturbation. Since the first and second conditions are not satisfied for most datasets in data classification, precise ICA reconstruction cannot be achieved. Under this circumstance, we observed that different rotation perturbations may result in different privacy guarantee and it is possible to find one rotation that is satisfactorily resilient to the enhanced ICA-based attacks. The following method defines how to evaluate the resilience against the enhanced ICA-based attacks.

Without loss of generality, we suppose that the level of confidence for an attack is primarily based on the PDF similarity between the two matched columns. Let O be the reconstruction of the original dataset X. $\Delta PDF(\mathbf{O}_i, \mathbf{X}_j)$ represents the PDF difference of the column i in O and the column j in X. Let $\{(1), (2), \ldots, (d)\}$ be a permutation of the sequence $\{1, 2, \ldots, d\}$, which means a match from the original column i to (i). An optimal match minimizes the sum of PDF differences of all pair of matched columns. We define the minimum privacy guarantee based on the optimal match as follows.

$$p^{min} = min\{\frac{1}{w_k} VoD(\mathbf{X}_k, \mathbf{O}_{(k)}), 1 \leq k \leq d\} \tag{27}$$

where $\{(1), (2), \ldots, (d)\} = argmin_{\{(1),(2),\ldots,(d)\}} \sum_{i=1}^{d} \Delta PDF(\mathbf{X}_i, \mathbf{O}_{(i)})$ Similarly, we can define the average privacy guarantee p^{avg} based on an optimal match between all columns as well.

With the above measures, we are able to estimate how resilient a rotation perturbation is to the ICA-based attacks that incorporate the knowledge of column statistics. We observed in experiments that, although the ICA method may effectively reduce the privacy guarantee for certain rotation perturbations, we can always find some rotation matrices so that they can provide satisfactory privacy guarantee to ICA-based attacks.

5.5.4 Attacks to Rotation Center

The basic rotation perturbation uses the origin as the rotation center. Therefore, the points around the origin will be still close to the origin after the perturbation, which leads to weaker privacy protection over these points. The attack to rotation center is another kind of naive estimation. We address this problem with random translation perturbation. The sophisticated attack to the enhanced perturbation would utilize the ICA technique. Therefore, we discuss this problem after we presented the ICA-based attack.

A random translation vector (matrix) has been defined earlier. Concretely, each dimensional value of the random translation vector $\mathbf{t}$ is uniformly drawn from the range [0, 1], so that the center hides in the normalized data space, resilient to estimation. There are two candidates for the extended perturbation.

$$\text{Transformation (1):} \qquad G(X) = R(X + \Psi)$$

or

$$\text{Transformation (2):} \qquad G(X) = RX + \Psi = R(X + R^{-1}\Psi)$$

It is easy to verify that $R^{-1}\Psi$ is also a translation matrix. Thus, the two are equivalent. We will use Transformation (2) in the complete version of geometric perturbation.

The effectiveness of random translation to protecting the rotation center is evaluated by how easy it is to estimate Ψ (or $R^{-1}\Psi$). One approach is again via ICA reconstruction. We assume that attackers know the basic column statistics for effective ICA-based attacks. Since translation just moves the mean of PDF function but preserves the shape of PDF, we can still find the matches by "PDF Alignment" and get the estimated R: $\hat{R}$. Then, an estimation to $\mathbf{t}$ can be done by the following steps.

Take Transformation (1) as example. Let P be the perturbed data. The estimate given by ICA is $\widehat{X + \Psi} = \hat{R}^{-1}P$. Suppose the original column i has the maximum and minimum values max_i and min_i, respectively, and $\hat{R}^{-1}P$ has max'_i and min'_i, respectively. As the process of ICA shows [63], the reconstruction may scale the original data column with some factor s, which can be estimated by $s \approx \frac{max'_i - min'_i}{max_i - min_i}$. Then, the attackers are able to estimate the translation matrix Ψ based on $\hat{R}^{-1}P$. First, the column i of $\hat{R}^{-1}P$ is scaled down to the same span of X by the factor s. Then,

the translation t_i for column i is estimated by

$$\hat{t}_i \approx min'_i \times s - min_i$$

Apparently, the quality of the estimated Ψ is dependent on the quality of ICA reconstruction. By optimizing the resilience to ICA-based attacks, Ψ will be well protected as well.

5.5.4.1 Effect to Model Accuracy

We have shown that random translation can effectively protect the rotation center from attacks. On the other hand, we also need to prove that this additional component will not seriously affect the model accuracy of the three categories of classifiers. Since translation does not change the distance relationship and hyperplane-based class boundary, it is easy to prove that kernel methods, linear classifiers, and SVM classifiers with radial basis function [57] will be invariant to translation.

However, translation does not preserve inner product. Therefore, it would be more complicated to directly prove the classifiers based on inner product, such as the SVM classifiers with polynomial kernels and sigmoid kernels. We will ignore the formal proofs here and show some results in experiments.

5.5.5 Distance-inference Attack

In the previous section, we have discussed naive estimation, ICA-based attacks, and attacks to rotation center. In the following discussion, we assume that, besides the information necessary to perform the discussed attacks, the attacker manages to get more knowledge about the original dataset: s/he also knows at least $d + 1$ original data records, $\{\mathbf{x}_1, \mathbf{x}_2, \ldots, \mathbf{x}_{d+1}\}$. S/he then tries to find the mapping between these points and their images in the perturbed dataset, denoted by $\{\mathbf{o}_1, \mathbf{o}_2, \ldots, \mathbf{o}_{d+1}\}$, to break the rotation and translation perturbation.

With the known points, it is possible to find their images in the perturbed data. Particularly, if a few ($\geq d + 1$) original points, such as the "outliers", are known, their images in the perturbed data can be found with high probability for low-dimensional small datasets (< 4 dimensions). With considerable cost, it is not impossible for higher dimensional larger datasets by simple exhaustive search. With the known mapping, the rotation R and translation $\mathbf{t}$ can be precisely calculated if only

the geometric perturbation $G(X) = RX + \Psi$ is applied. Therefore, the threat is substantial to the basic geometric perturbation.

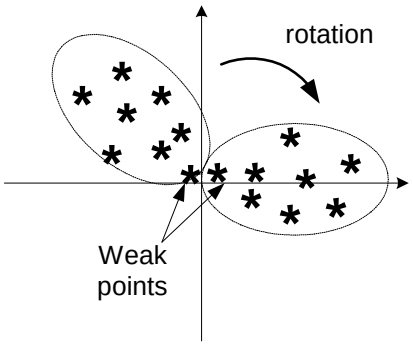

Figure 77: Weak points around the default rotation center.

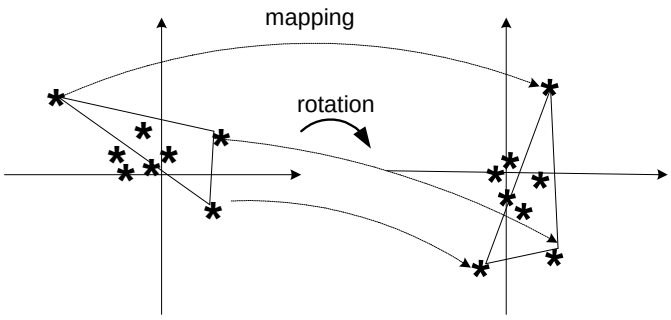

Figure 78: Using known points and distance relationship to infer the rotation matrix.

In order to protect from the distance-inference attack, we introduce an additional noise component $\Delta = [\delta_1, \delta_2, \ldots, \delta_N]$, δ_i is d-dimensional Gaussian random vector, to form the complete version of geometric perturbation, $G(X) = RX + \Psi + \Delta$. Under this perturbation, we analyze how the attacker can estimate the original data with the known points and mappings to decide how intense the noise δ_i should be.

There are two possible scenarios. In the first scenario, the attacker does not know the exact matching between the known original data records and their images in the perturbed data. The attacker has to figure out the accurate matches with the distance information. However, because the distance relationship has been disturbed, there is no confidence guarantee with any plausible matches.

In the second scenario, we assume that the attacker can get (or guess) the right mapping between the original points and their images in the perturbed data: $\{\mathbf{x}_1, \mathbf{x}_2, \ldots, \mathbf{x}_{d+1}\} \rightarrow \{\mathbf{o}_1, \mathbf{o}_2, \ldots, \mathbf{o}_{d+1}\}$, where $\mathbf{o}_i$ is perturbed by the noise component, i.e., $\mathbf{o}_i = R\mathbf{x}_i + \mathbf{t} + \delta_i$. Suppose δ_i are independently drawn from Gaussian distribution $N(0, \sigma^2)$. The attacker can only make the following best effect to estimate R and $\mathbf{t}$.

Step 1. R is estimated as follows. The translation vector $\mathbf{t}$ can be canceled from the perturbation and we get d equations: $\mathbf{o}_i - \mathbf{o}_{d+1} = R(\mathbf{x}_i - \mathbf{x}_{d+1}) + \delta_i - \delta_{d+1}$, $1 \leq i \leq d$. Let $\bar{O} = [\mathbf{o}_1 - \mathbf{o}_{d+1}, \mathbf{o}_2 - \mathbf{o}_{d+1}, \ldots, \mathbf{o}_d - \mathbf{o}_{d+1}]$, $\bar{X} = [\mathbf{x}_1 - \mathbf{x}_{d+1}, \mathbf{x}_2 - \mathbf{x}_{d+1}, \ldots, \mathbf{x}_d - \mathbf{x}_{d+1}]$, and $\bar{\Delta} = [\delta_1 - \delta_{d+1}, \delta_2 -$

$\delta_{d+1}, \ldots, \delta_d - \delta_{d+1}]$. The equations are unified to $\bar{O} = R\bar{X} + \bar{\Delta}$, and estimating R becomes a linear regression problem. Let $\bar{X}^t$ be the transpose of $\bar{X}$. It follows that the minimum variance unbiased estimator for R is

$$\hat{R} = \bar{O}\bar{X}^t(\bar{X}^t\bar{X})^{-1}$$

[83, 57].

Step 2. With $\hat{R}$, the translation vector $\mathbf{t}$ can also be estimated. Since $\mathbf{o}_i - R\mathbf{x}_i - \delta_i = \mathbf{t}$ and δ_i has mean value 0, with $\hat{R}$ we have the estimate of $\mathbf{t}$ as

$$\begin{aligned} \hat{\mathbf{t}} &= \frac{1}{d+1}\{\sum_{i=1}^{d+1}(\mathbf{o}_i - \hat{R}\mathbf{x}_i) - \sum_{i=1}^{d+1}\delta_i\} \\ &\approx \frac{1}{d+1}\sum_{i=1}^{d+1}(\mathbf{o}_i - \hat{R}\mathbf{x}_i) \end{aligned}$$

. However, $\hat{\mathbf{t}}$ will have considerable variance brought by the components $\hat{R}$ and δ_i.

Step 3. With $\hat{R}$ and $\hat{\mathbf{t}}$, the original data X can be estimated. As $O = RX + \Psi + \Delta$, using the estimators $\hat{R}$ and $\hat{\Psi} = [\hat{\mathbf{t}}, \ldots, \hat{\mathbf{t}}]$, we get $\hat{X} = \hat{R}^{-}1(O - \hat{\Psi})$. Due to the variance introduced by $\hat{R}$, $\hat{\Psi}$, and Δ, in practice the attacker may actually need more samples to perform several runs to get several estimated $\hat{X}_i$, and then uses the mean of $\hat{X}_i$ as the final estimate.

With the above estimation (attacking) process, we are able to simulate and get the actual privacy guarantee to the attack. Here, the unified privacy metric can be calculated with $Var\{\hat{X} - X\}$. When the noise component is introduced, we may have to sacrifice some model accuracy for gaining the stronger privacy protection. We will further study the relationship between the noise level, the privacy guarantee, and the model accuracy in experiments.

5.5.6 Other Potential Attacks

We have studied four kinds of attacks, according to the different levels of knowledge that an attacker may have. The distance-inference attack presents an extreme case that the attacker can know some specific points in the original dataset and their images in the perturbed dataset. AK_ICA [55] investigates a scenario that may also rarely happen in practice. It assumes the attacker can know a significant amount ($\gg d+1$) of the original data points, although the amount is still relatively small compared to the total number of records. These known points might contain significant information, such as the distribution, the covariance matrix of the original dataset. Therefore, theoretically

this information can be used to model the original data. Typical methods, such as PCA [57] and ICA, can then be used to reconstruct the original dataset with the approximate information from both the known points and the perturbed data. However, unless the known points can approximately describe the distribution of original dataset, these methods will be not so effective. Furthermore, with the random translation and the additional noise component, the information from the perturbed dataset might be inconsistent with that from the original dataset. As a result, such methods would be ineffective on the full version of geometric perturbation, even though a considerable amount of original points are known. Further studies will be performed on such kind of attacks.

5.6 Randomized Algorithm for Finding Resilient Perturbations

We have analyzed the related inference attacks with the help of multidimensional privacy evaluation model, which allows us to design an algorithm to choose a geometric perturbation resilient to these inference attacks. Considering that a determined algorithm in perturbation optimization may provide extra clue to privacy attackers, we try to randomly optimize the perturbation so that the attacker cannot inference any additional information from the algorithm itself.

Algorithm 1 runs in a given number of iterations, aiming at locally maximizing the minimum privacy guarantee. Initially, a random translation is selected. In each iteration, the algorithm randomly generates a rotation matrix. Local swapping-based optimization of rotation is then applied to find a better rotation matrix against naive estimation, which is then tested by the ICA reconstruction method by the methods defined in Section 5.5.3. The rotation matrix is accepted as the currently best rotation if it provides higher minimum privacy guarantee than the previous rotations. After the limited number of iterations, finally, the noise component is appended to the perturbation, so that the distance-inference attack cannot reduce the privacy guarantee to a safety level ϕ, e.g., $\phi = 0.2$. Algorithm 1 outputs the rotation matrix R_t, the random translation vector $\mathbf{t}$, the noise level σ^2, and the minimum privacy guarantee. If the privacy guarantee is lower than an anticipated threshold, the data owner can select not to release the data.

Note that the distance-inference attack is optimized separately. The additional noise component will further reduce the effectiveness of naive estimation and ICA-based attack. Therefore, the actual privacy guarantee will be higher than the evaluated result.

Algorithm 1 Finding a resilient perturbation ($X_{d\times N}$, **w**, ϕ, m)

Input: $X_{d\times N}$:the original dataset, **w**: weights of attributes in privacy evaluation, ϕ: the expected privacy guarantee in terms of distance-inference attack, m: the number of iterations.
Output: R_t: the selected rotation matrix, Ψ: the random translation, σ^2: the noise level, p: privacy quality
calculate the covariance matrix C of X;
$p = 0$, and randomly generate the translation Ψ;
for Each iteration **do**
  randomly generate a rotation matrix R;
  swapping the rows of R to get R_1, which maximizes $min_{1\le i\le d}\{\frac{1}{w_i}(Cov(R_1X - X)_{(i,i)}\}$;
  $p_0 =$ the privacy guarantee of R_1, $p_1 = 0$;
  if $p_0 > p$ **then**
    generate O with ICA;
    scale the columns in O with the maximum/minimum values of original columns;
    $\{(1), (2), \ldots, (d)\} = argmin_{\{(1),(2),\ldots,(d)\}} \sum_{i=1}^{d} \Delta PDF(X_i, O_{(i)})$
    $p_1 = min_{1\le k\le d} \frac{1}{w_k} VoD(X_k, O_{(k)})$
  end if
  if $p < \min(p_0, p_1)$ **then**
    $p = \min(p_0, p_1)$, $R_t = R_1$;
  end if
end for
$p_2 =$ the privacy guarantee to the distance-inference attack with the perturbation $G(X) = R_tX + \Psi + \Delta$.
Tune the noise level σ^2, so that $p_2 \ge \phi$

5.7 Experiments

We design four sets of experiments to evaluate the geometric perturbation approach. The first set is designed to show that the discussed classifiers are invariant to rotations, and even partially invariant to general linear transformation. The second set shows the optimization of the privacy guarantee in the basic geometric perturbation (without noise addition) in terms of naive-inference attack and ICA-based attack. Note the privacy guarantee can be higher if the noise component is considered. In the third set of experiments, we study the threat of distance-inference attacks and the relationship between the additional noise component of the geometric perturbation, the privacy guarantee, and the model accuracy. Finally, we compare the privacy guarantee provided by our random geometric perturbation (without the noise component) and another multidimensional perturbation − condensation approach. All datasets used in the experiments can be found in UCI machine learning database [3].

5.7.1 Rotation-invariant Classifiers

In this experiment, we verify the invariance property of several classifiers discussed in section 5.4.2. Three classifiers: KNN classifier, SVM classifier with RBF kernel, and perceptron, are used as the

[3] http://www.ics.uci.edu/∼mlearn/Machine-Learning.html

Table 8: Experimental result on rotation transformation

Dataset	N	d	k	KNN		SVM(RBF)		Perceptron	
				orig	R	orig	R	orig	R
breast-w	699	10	2	97.6	-0.5 ± 0.3	97.2	0 ± 0	89.1	-4.9 ± 1.2
credit-a	690	14	2	82.7	$+0.2 \pm 0.8$	85.5	0 ± 0	64.6	$+4.7 \pm 1.5$
credit-g	1000	24	2	72.1	$+1.2 \pm 0.9$	76.3	0 ± 0	70.1	-0.1 ± 0
diabetes	768	8	2	73.3	$+0.4 \pm 0.5$	77.3	0 ± 0	66.6	-4.5 ± 0.8
ecoli	336	7	8	85.1	$+0.2 \pm 0.8$	78.6	0 ± 0	-	-
heart	270	13	2	78.9	$+2.1 \pm 0.5$	84.8	0 ± 0	67.4	-0.4 ± 1.0
hepatitis	155	19	2	80.8	$+1.8 \pm 1.5$	79.4	0 ± 0	79.4	-0.3 ± 0.8
ionosphere	351	34	2	86.4	$+0.5 \pm 0.6$	89.7	0 ± 0	66.9	-1.8 ± 0.6
iris	150	4	3	94.6	$+1.2 \pm 0.4$	96.7	0 ± 0	-	-
tic-tac-toe	958	9	2	99.0	-0.3 ± 0.4	70.4	0 ± 0	56.6	$+8.0 \pm 0.6$
votes	435	16	2	92.5	$+0.4 \pm 0.4$	95.6	0 ± 0	60.3	-2.8 ± 1.3
wine	178	13	3	98.3	-0.6 ± 0.5	98.9	0 ± 0	-	-

representatives of the three types of classifiers, respectively.

Each dataset is randomly rotated 10 times with different rotation matrices. Each of the 10 resultant datasets is used to train and cross-validate the classifiers. The reported numbers are the average of the 10 testing results. We calculate the difference of performance, i.e., model accuracy, between the classifier trained with the original data and those trained with the rotated data.

In the table 8, 'orig' is the classifier accuracy to the original datasets, 'R' denotes the result of the classifiers trained with rotated data, and the numbers in 'R' columns are the performance difference between the classifiers trained with original and rotated data, for example, "-1.0 ± 0.2" means that the classifiers trained with the rotated data have the accuracy rate 1.0% lower than the original classifier on average, and the standard deviation is 0.2%. We use single-perceptron classifiers in the experiment. Therefore, the datasets having more than two classes, such as "E.Coli", "Iris" and "Wine" datasets, are not evaluated for perceptron classifier.

From table 8, we can see that both types of transformations almost do not change the accuracy rate of the classifiers (with very small difference and standard deviation). Rotation surprisingly increases the accuracy of KNN classifiers a little bit, while keeps the SVM (RBF) unchanged. The accuracy of Perceptron classifiers fluctuates from dataset to dataset. It is also surprising that the accuracy on tic-tac-toe data increases significantly after rotation. This fluctuation of accuracy may caused by Perceptron's training process.

5.7.2 Perturbation Optimization against Naive Estimation and ICA-based Attack

We run the randomized optimization algorithm and show how effective it can generate resilient perturbations. Each column in the experimental dataset is considered equally important in privacy evaluation, thus, the weights are not included in evaluation.

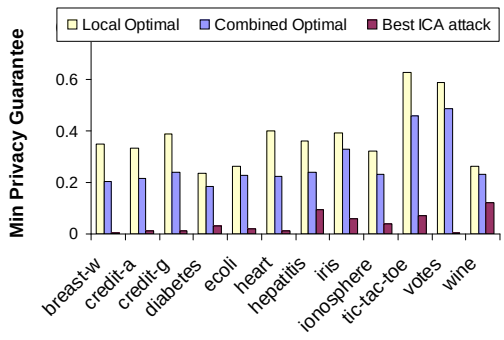

Figure 79: Minimum privacy guarantee generated by local optimization, combined optimization, and the performance of ICA-based attack.

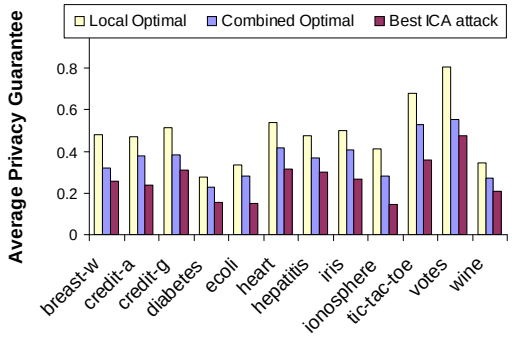

Figure 80: Average privacy guarantee generated by local optimization, combined optimization, and the performance of ICA-based attack.

Figure 79 and 80 summarize the evaluation of privacy quality on experimental datasets. The results are obtained in 50 iterations with the optimization algorithm described in Section 5.6. The "Local Optimal" represents the locally optimized minimum privacy guarantee addressing naive estimation. "Best ICA attack" is the worst perturbation that gives the best ICA attack performance, i.e., getting the lowest privacy guarantee among the 50 perturbations. "Combined Optimal"is the combined optimization result given by Algorithm 1 at the end of 50 iterations. The above values are all standard deviation of the difference between the perturbed dataset (or the estimated dataset) and the original dataset. The "Local Optimal" values can often reach a high level after 50 iterations, which means that the swapping method is very efficient in locally optimizing the privacy quality. The best ICA attacks often result very low privacy guarantee, which means some rotation perturbations are weak to ICA-based attacks. "Combined Optimal" values are much higher than the corresponding ICA-based attacks, which supports our conjecture that we can always find one perturbation that is sufficiently resilient to ICA-based attacks if the four conditions for perfect ICA reconstruction are not satisfied.

We also show the detail in the course of optimization for two datasets "Diabetes" and "Votes"

in Figure 81 and 82, respectively. For both datasets, since the lowest privacy guarantees reduced by ICA-based attacks are lower than the result of swapping-based optimization, the combined optimal result is located in between the curves of best ICA-attacks and the best local optimization result. In the case that ICA-based attacks are not effective, i.e., the "best ICA attack" is higher than local optimization curve, we take the local optimization curve as the combined optimal result.

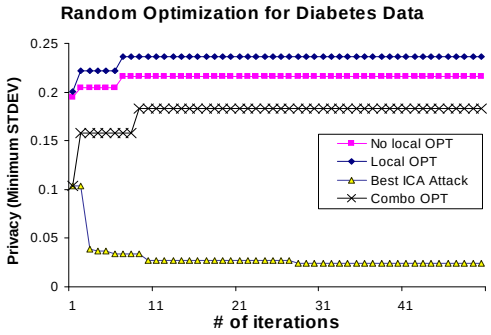

Figure 81: Optimization of perturbation for Diabetes data.

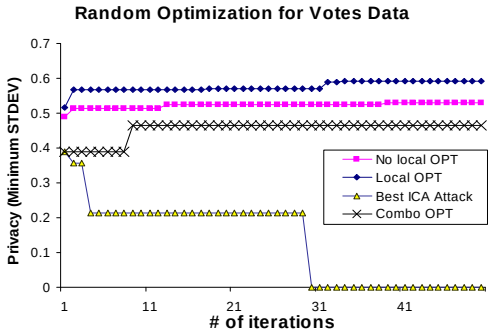

Figure 82: Optimization of perturbation for Votes data.

5.7.3 Effectiveness of Translation Perturbation

In this set of experiments, firstly, we show that it is ineffective to estimate the rotation center if the translation perturbation is appended. As we have mentioned, if the translation vector could be precisely estimated, the rotation center would be exposed. We applied the ICA-based attack to rotation center that is described in Section 5.5.4. The data in Figure 83 shows $stdev(\hat{\mathbf{t}} - \mathbf{t})$ which is equivalent to the VoD used in multidimensional privacy evaluation model. Compared to the range of the elements in $\mathbf{t} - [0, 1]$, the standard deviations are quite large, so we can conclude that random translation will also be safe to attacks, if we have optimized the resilience of rotation perturbation in terms of ICA-based attacks.

Secondly, we show that the two classifiers, SVM with polynomial kernel, and SVM with sigmoid kernel, are also invariant to translation transformation. Table 9 lists the experimental result on the 12 datasets. We randomly translate each dataset for ten times. The result is the average of the ten runs. For most datasets, the result shows zero or tiny deviation from the standard model accuracy.

Table 9: Experimental result on random translation

Dataset	SVM(polynomial)		SVM(sigmoid)	
	orig	Tr	orig	Tr
breast-w	96.6	0 ± 0	65.5	0 ± 0
credit-a	88.7	0 ± 0	55.5	0 ± 0
credit-g	87.3	-0.4 ± 0.4	70	0 ± 0
diabetes	78.5	0 ± 0.3	65.1	0 ± 0
ecoli	89.9	-0.1 ± 0.5	42.6	0 ± 0
heart	91.1	-0.2 ± 0.2	55.6	0 ± 0
hepatitis	96.7	-0.4 ± 0.3	79.4	0 ± 0
ionosphere	98	$+0.3 \pm 0$	63.5	$+0.6 \pm 0$
iris	97.3	0 ± 0	29.3	-1.8 ± 0.4
tic-tac-toe	100	0 ± 0	65.3	0 ± 0
votes	99.2	$+0.2 \pm 0.1$	65.5	-4.7 ± 0.6
wine	100	0 ± 0	39.9	0 ± 0

5.7.4 Tradeoffs in Terms of Distance-inference Attack

We use the geometric perturbation with random noise component : $G(X) = RX + \Psi + \Delta$, to address the potential distance-inference attacks. From the formal analysis, we know that the noise component Δ can significantly affect the accuracy of distance-inference attack, thus provide certain privacy guarantee. Intuitively, the higher the noise level is, the better the privacy guarantee. However, with the increasing noise level, the model accuracy could be affected, too. In this set of experiments, we first study the relationship between the noise level, represented by the variance σ^2, and the privacy guarantee, as well as between the noise level and the model accuracy, with three datasets "Diabetes", "Iris", and "Votes". Then, we compare the privacy guarantee and the model accuracy for all of the experimental datasets at certain noise level ($\sigma = 0.1$).

Figure 84 shows, if the attack described in Section 5.5.5 is addressed with the noise component, the privacy guarantee increases with the increase of noise level. At the noise level $\sigma = 0.1$, the privacy guarantee is almost above 0.2. However, Figure 85 and 86 show the decreasing trend of accuracy change for KNN classifier and SVM (RBF kernel) classifier, respectively. With the noise level lower than 0.1, the accuracy of both classifiers is only reduced less than 6%, which is quite acceptable.

We summarize the privacy guarantees at the noise level 0.1 for all experimental datasets [4] in

[4] "Ionosphere" is not included because any combination of known d records results in a singular matrix. Therefore, the attack described in Section 5.5.5 does not work.

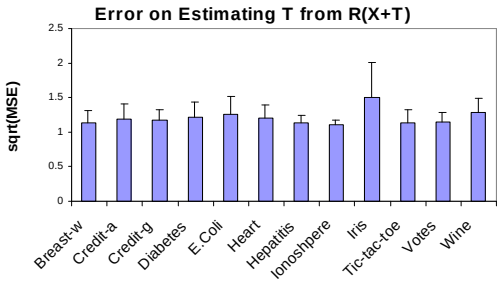

Figure 83: Resilience to the attack to random translation

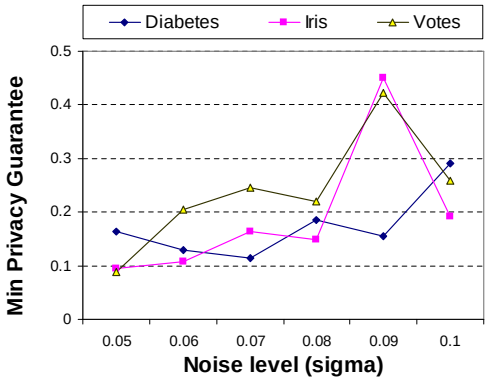

Figure 84: The change of minimum privacy guarantee vs. the increase of noise level for the three datasets.

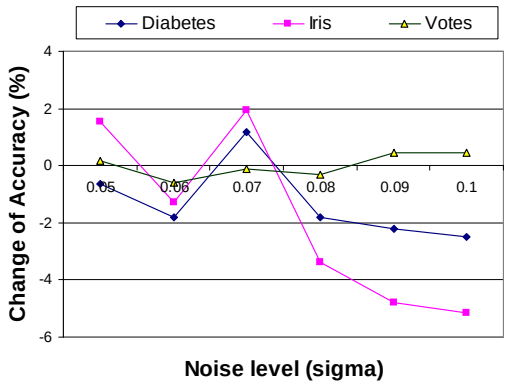

Figure 85: The change of accuracy of KNN classifier vs. the increase of noise level.

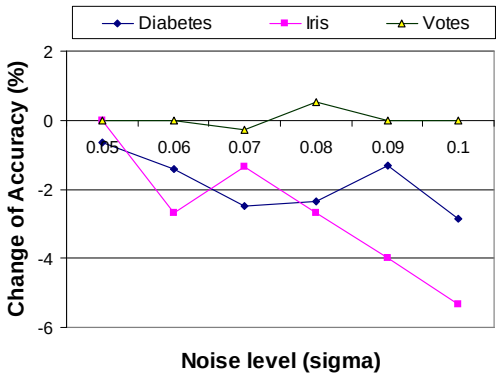

Figure 86: The change of accuracy of SVM(RBF) classifier vs. the increase of noise level.

Figure 87, and also the change of model accuracy for KNN, SVM(RBF), and Perceptron in Figure 88. The positive accuracy differences indicate that the perturbation increases the accuracy in some cases. Except the boolean datasets "Votes" and "Tic-tac-toe" are quite sensitive to the noise component, most of the results show that, with small noise addition, we can get satisfactory privacy guarantee with small sacrifice of model accuracy.

5.7.5 Geometric Perturbation Approach vs. Condensation Approach.

We design a simple algorithm to estimate the privacy quality of condensation approach. As we mentioned, since its perturbation part is done within the KNN neighbors, it is highly possible that the perturbed data is in the KNN neighbors of the original data too. For each record in the perturbed

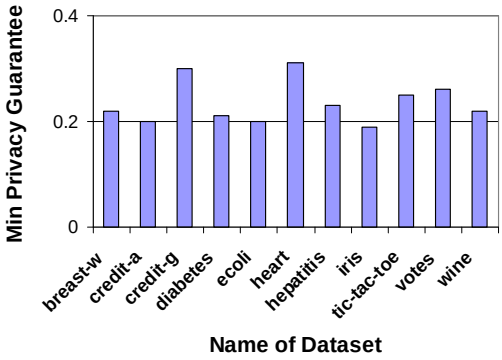

Figure 87: Minimum privacy guarantee at the noise level $\sigma = 0.1$

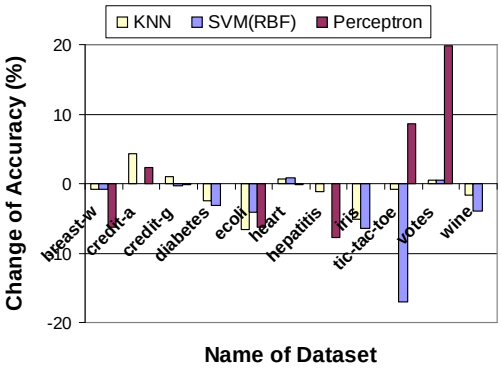

Figure 88: The change of model accuracy at the noise level $\sigma = 0.1$

dataset, we try to find the nearest neighbor in the original data. By comparing the difference between the perturbed data and its nearest neighbor in the original data, we can approximately measure the privacy quality of condensation approach, which is a typical analysis for naive-inference attacks.

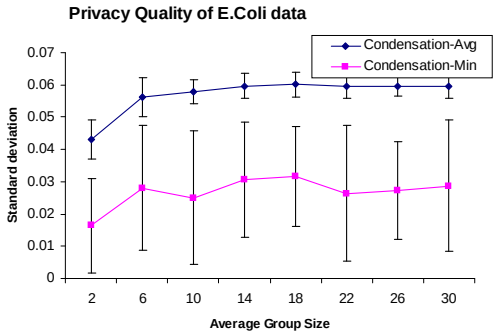

Figure 89: Privacy quality of condensation approach on E.Coli data.

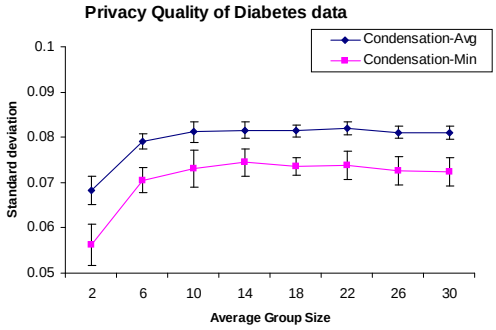

Figure 90: Privacy quality of condensation approach on Diabetes data.

Intuitively, the better locality the KNN perturbation is, the better the condensation approach can preserve the information, but the worse the privacy guarantee is. Figure 89 and 90 show the relationship between the size of condensation group and the privacy quality on "E.Coli" and "Diabetes" datasets. It was demonstrated in the paper [2] that the accuracy of classifier becomes stable with the increase of the size of condensation group. However, we observed that the privacy quality generally stays low, no matter how the condensation size changes. Experiment on both datasets shows the minimum privacy guarantees are very low, neither are the average privacy levels. We also observed that the minimum privacy is 0 for "Ionosphere" data, which happens to the column that contains the

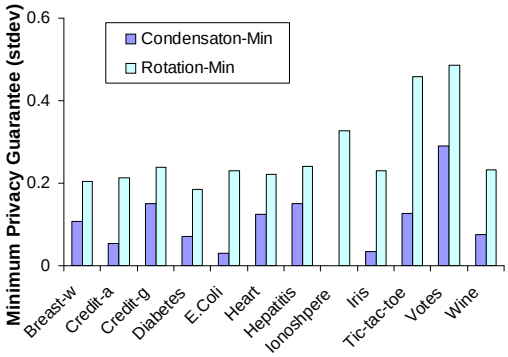

Figure 91: Comparison on minimum privacy level.

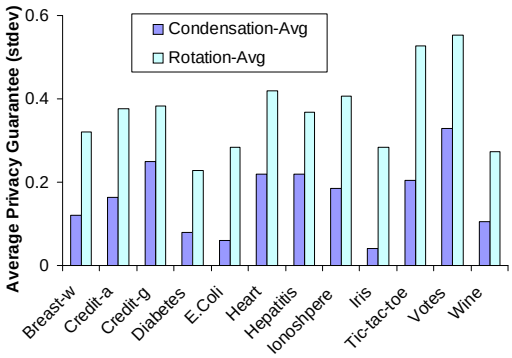

Figure 92: Comparison on average privacy level.

identical value. The condensation approach seems not working for such cases at all. Supported by the other two Figures (91 and 92), we can conclude that the condensation approach only provides weak privacy protection and we cannot possibly adjust the perturbation to meet any higher privacy requirement.

While our geometric approach well preserves the model-specific information for classification, it also provides much higher privacy quality than the condensation approach. Figure 91 and 92 shows the comparison on the minimum privacy guarantee and the average privacy guarantee of the two approaches, respectively. The numbers for geometric perturbation are the results generated by the random optimization algorithm in 50 iterations. We see that the our approach can generally provide much higher privacy guarantee than the condensation approach.

CHAPTER VI

PRIVACY-PRESERVING MULTIPARTY COLLABORATIVE DATA CLASSIFICATION WITH GEOMETRIC PERTURBATION

We have shown that geometric data perturbation has unique benefits in data classification mining. First, it can be used for multiple popular data classification models, such as kernel methods, linear classifiers, and SVM classifiers, without any further development of new classification algorithms. Second, balancing the loss of data quality and the loss of privacy becomes relatively easy in geometric perturbation. It can provide high privacy guarantee with little loss of model accuracy for the discussed classifiers. Despite these benefits, we identify that "unifying data spaces" is the basic challenge in applying geometric perturbation to *multiparty* collaborative privacy-preserving data classification. In this chapter, we propose three protocols to address the problem of unifying data spaces, discuss the advantages and disadvantages of each protocol, and experimentally evaluate the relationship between the related factors for the three protocols. We show that with these protocols we are able to efficiently extend the unique benefits of geometric perturbation technique to multiparty collaborative privacy-preserving data classification.

This chapter is organized as follows. In section 6.1, we introduce the problem of privacy-preserving multiparty collaborative data classification and the challenges of applying geometric perturbation to this application. Then, we will describe the data-mining-service oriented framework (Section 6.2.2) and formally define the data space (perturbation) unification problem (Section 6.2.3). Two approaches, including four protocols, for unifying perturbations are discussed in Section 6.3. Finally, we report some experimental results and discuss the balance between multiple factors in the three protocols.

6.1 Multiparty Collaborative Data Classification

With the rapidly growing applications in Internet and grid computing, large datasets have been generated and spread over distributed sites. There is an increasing demand on collaborative mining over

the distributed data stores to find the patterns or rules that benefit all of the participants. For example, multiple retailer stores in the same business section want to pool their data together to determine the characteristics of customer purchases. Cancer research institutes in different geographical areas need to collaboratively find the environmental factors related to certain type of cancer. However, these distributed datasets could also contain sensitive information, such as business sales data and patient personal information. Therefore, there raises the challenge: how to collaboratively find the useful data model (classification model in this chapter) that can benefit all participants, while each participant's sensitive information is not breached?

Data mining tasks often require considerable computing resource. Most importantly, these tasks have to be done by well trained data analysis experts in order to get meaningful models. However, in many cases, the parties interested in collaborative mining may not have such resources, and it is often inefficient to recruit experts and purchase additional expensive computing resources individually. Meanwhile, many companies and research institutes have abundant computing resources and expertise in data analysis and data mining, which can serve as a data mining service provider. Therefore, the participants (the data providers) of collaborative distributed mining can submit their datasets to the designated data mining service provider (the data miner) for mining the commonly interested models. Such a mining-service based framework has become a popular framework recently [4, 113]. In this chapter, we will study the issues of multiparty collaborative data classification with *geometric data perturbation* under this mining-service based framework.

We have proposed the geometric data perturbation technique for single data owner publishing privacy sensitive data [25, 24]. Geometric data perturbation has unique benefits for privacy-preserving data classification. First, a bunch of popular classifiers, such as kernel methods (including KNN classifer), linear classifiers, and SVM classifiers, can use geometrically perturbed data directly, so that the classifiers trained with the perturbed data have almost the same accuracy with those trained with the original data. Second, since any geometric data perturbation preserves the model accuracy, for single data provider we need only to find one perturbation that can provide satisfactory privacy guarantee. This has significantly reduced the complexity in balancing data quality and privacy guarantee in data perturbation [4, 40].

However, when geometric data perturbation is applied to multiparty collaborative data classification under the mining-service framework, there is a particular challenge – how to unify the perturbations used by different data providers, so that each party still gets satisfactory privacy guarantee while the data quality is also well preserved. There are several issues involved in unifying perturbations. First, if parties need to interact with each other, the interaction may reveal some information that compromises parties' privacy. We should identify these potential threats. Second, in order to preserve satisfactory privacy guarantee, we may need to sacrifice some data quality or pay some cost in unifying perturbations. It is, thus, critical to understand the relationship between the three aspects: the data privacy, the data quality, and the efficiency, of a concrete protocol, in order to efficiently unifying perturbations with high overall privacy guarantee.

6.1.1 Scope and Contributions

In this chapter, we address the challenges in multiparty collaborative privacy-preserving data mining with geometric perturbation, and propose two protocols to securely unifying the perturbations. Concretely, it has the following major contributions.

1. We formally define the problem of unifying perturbations for multiple data providers using geometric perturbation.

2. We propose two approaches (including four protocols) to unifying data spaces: 1) ranking perturbations to find one best perturbation, and 2) securely transforming the perturbations to one randomly generated perturbation.

3. We study the major factors that can significantly affect the protocols, and discuss how to balance them in order to achieve better efficiency, privacy guarantee, and model accuracy.

6.2 Preliminary

In this section, we introduce the basic concepts in geometric perturbation, the setting of the mining-service based framework for multiparty collaborative mining, and the issues involved in applying geometric perturbation to this framework.

6.2.1 Geometric Perturbation

Geometric perturbation includes the combination of random rotation perturbation and random translation perturbation. Without loss of generality, it can be represented as $G(X) = RX + \Psi + \Delta$, where $X_{d\times N}$ is the original dataset with N rows and d columns, R is an orthogonal matrix (rotation matrix), and Ψ is a translation matrix $\Psi = \mathbf{t} \times \mathbf{1}^t$, $\mathbf{t} = [t_1, t_2, \ldots, t_d]^t$ $(0 \leq t_i \leq 1, 1 \leq i \leq d)$, and $\mathbf{1} = [1, 1, \ldots, 1]^t$. Δ is an iid noise matrix with certain distribution. Since a noise component with $N(0, 0.1^2)$ can give satisfactory privacy guarantee in terms of corresponding distance-inference attack [24], without loss of generality, we use only $(R, \mathbf{t})$ to represent a perturbation in this chapter. Geometric perturbation is specially designed for a family of "geometric transformation invariant classifiers". These classifiers, if trained over the perturbed data, will have similar accuracy to those trained over the original data. Particularly, we have proved that this family of classifiers include KNN, kernel methods, linear classifiers, and SVM classifiers with the commonly used kernels, which are all popularly used in real-world applications. Since there are numerous geometric perturbations for a given dataset, which all can give the similar model accuracy for these classifiers, the problem becomes how to find a "good" perturbation that can provide high privacy guarantee.

Solving the problem of finding good geometric perturbations requires us to answer the following questions: "how to define the privacy guarantee for the data perturbed by geometric perturbation?" and "how to design the optimization algorithm that generate perturbations resilient to attacks?" The geometric perturbation method addresses these questions with the following three contributions.

Multi-column privacy evaluation: In multidimensional data perturbation, such as geometric perturbation, the privacy guarantee of an individual data column is correlated to that of other columns, so we need to optimize the privacy guarantee of all columns together, not individually handling them like in randomization approach [4]. We propose a generic evaluation model to compose the unified column privacy metric. This unified column privacy metric extends the commonly used variance-based metric [4], considering the columns on the same base. Let C_i represent the column i of the original dataset and O_i be the observed column i, which is the perturbed data or is reconstructed from the perturbed data by potential attacks. We define the privacy guarantee of column i as p_i, $p_i = stdev(O_i - C_i)$, where "stdev" is the standard deviation of the difference. Let

w_i be the significance of the column i in privacy protection. Intuitively, the higher the significance is in privacy, the proportionally higher the privacy guarantee is required. Therefore, p_i/w_i will be appropriate for a weighted privacy guarantee. Since all columns have been normalized to the same range, we can approximately compare the privacy guarantees crossing different columns, with the following two metrics:

Minimum Privacy Guarantee: $\min\{\frac{p_1}{w_1}, \frac{p_2}{w_2}, \ldots, \frac{p_d}{w_d}\}$

Average Privacy Guarantee: $\frac{1}{d}\sum_{i=1}^{d}\frac{p_i}{w_i}$

We can use the two multi-column metrics to evaluate the privacy quality of the multidimensional perturbation. In experiments, all column weights are equal and we simply use p_i instead. We also choose to evaluate only the "Minimum Privacy Guarantee" in this chapter. Therefore, by default the privacy guarantee is Minimum Privacy Guarantee. Under the multiparty scenario, the privacy guarantees are locally evaluated with the multi-column privacy evaluation model by each party. Therefore, in experiments, we also consider the minimum/average privacy guarantee over all parties.

Attack analysis: According to different levels of attacking knowledge, we categorize the possible attacks to geometric perturbation into three categories. 1) Naive-estimation attack. The attackers have no additional knowledge about the original data, so they simply estimate the original records from the perturbed data. 2) Independent Component Analysis (ICA) based attack. If the attackers know some column statistics, such as the maximum/minimum values and the probability density function of each column, they can apply ICA technique to possibly reconstruct the original dataset. However, the effectiveness of ICA-based attack is determined by the property of the original dataset, which can be well-predicted and tested. If the conditions for effective ICA are not well satisfied, we can always find one perturbation that is sufficiently resilient to the ICA-based attack. 3) Distance-inference attack. If the attackers know some original points and their maps in the perturbed dataset, they can use this knowledge to break geometric perturbation if only random rotation and translation are applied. The noise component Δ is used to perturb the distance relationship. Experimental result shows that with low noise intensity, geometric perturbation is resilient to distance-inference attack, while the model accuracy is also well preserved.

Randomized perturbation optimization: Based on the analysis of attacks and the multi-column privacy evaluation model, an algorithm is designed to find a locally optimized perturbation

that is resilient to the three levels of attacks. The optimization process is also randomized so that the attackers cannot attack the algorithm. In collaborative privacy preserving data classification, the randomized optimization is used to find a resilient perturbation locally by each data provider.

6.2.2 Multiparty Mining-Service Based Framework

Service Computing is becoming a major paradigm in distributed computing and business processing. Since data mining is a resource-intensive task, involving highly centralized expertise and computing power, it could become an important service supported by the companies or research institutes that have abundant resources. In multiparty collaborative data mining, participants want to share their data to find valuable global models. However, they can also be competitive and do not want their own sensitive information in the shared data to be breached. Therefore, the individual privacy has to well preserved in order to make the collaboration smoothly happen.

Figure 93 shows the service-based framework for collaborative multiparty data mining. The data-mining service provider (SP) is a party independent of the data providers. The data providers (DP) contribute their own datasets and hope to find models built on the pooled data. The mined models are shared by the data providers. It is possible that a trusted third party (TP) also attends the mining, and malicious attackers will try to attack the data flows.

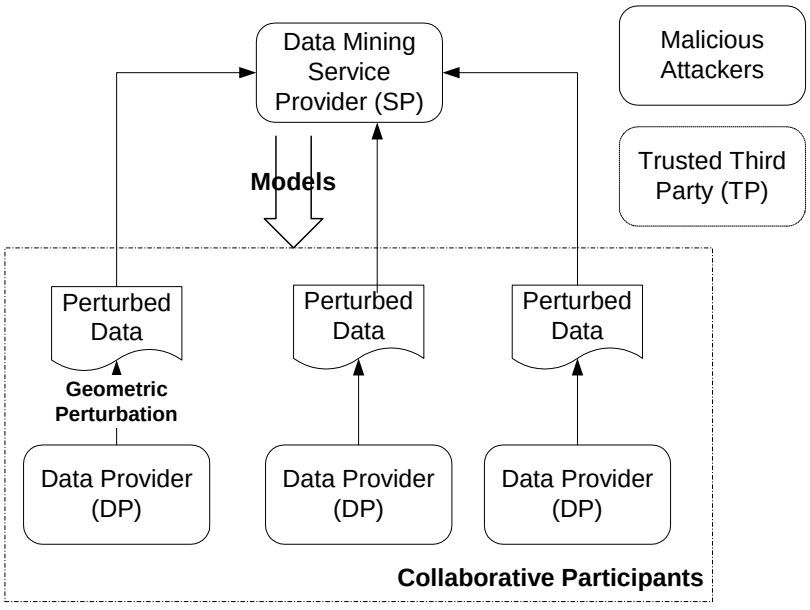

Figure 93: Service-based multiparty privacy-preserving data mining.

We consider that the multiparty protocols discussed in this chapter involve only semi-honest parties (DPs and SP). The semi-honest assumption states that the parties attending the protocol are not malicious parties: they will exactly follow the protocol and do not change the intermediate results. We also consider the following assumptions are also reasonable.

- No data provider deliberately exposes her own data or perturbation parameters to other parties or the malicious attackers.
- The data miner will not deliberately expose any received data to other parties or the malicious attackers.

However, the parties may be curious about other's sensitive information, trying to breach other's privacy with the data and information they receive in the protocol. In addition, the communication channels are also not secure, and may under attacks by curious parties and malicious attackers.

6.2.3 Challenges for Multiparty Geometric Perturbation

A geometric perturbation consists of random geometric transformations and a noise component. The random geometric transformations provide the major protection to data privacy (if the possibility of reveal original data does not exist, the noise component can be removed). A geometric transformation changes the coordinates of the data points, or in other words, it changes the entire coordinate system. Let the base vectors $V_0 = [\mathbf{e}_1, \mathbf{e}_2, \ldots, \mathbf{e}_d] = I_{d\times d}$ construct the original vector space V_0. By using the geometric perturbation G_i, we actually transform the vector space to V_i. Without loss of generality, we use geometric perturbation G_i to represent the transformed vector space V_i, and "data space" in this chapter is equivalent to "vector space". Let the sub-datasets $\{X_1, X_2, \ldots, X_k\}$ in V_0 held by the k data providers DP_i, respectively. The pooled dataset X in V_0 is the union of X_i, $X = \bigcup_{i=1}^{k} X_i$, $X_i \cap X_j = \emptyset, i \neq j$. Correspondingly, the data mining models trained with any of the subsets will be in the original space, too. Let G_i be the transformation used by the data provider i. We need to clarify the following concepts in terms of the definition of data space.

- If $G_i \neq G_j$, it is not meaningful to pool the two transformed datasets $G_i(X_i)$ and $G_j(X_j)$ together, since they are in the different data spaces.

- Correspondingly, let the classification model M_i trained with $G_i(X_i)$, and M_j with $G_j(X_j)$. M_i and M_j are not comparable.

Therefore, if multiple distributed data stores are perturbed into different data spaces, respectively, it is necessary to unify them into one data space (i.e., one perturbation) so that classification models with the pooled data can be validly built, analyzed, and used.

There can be several methods to unify the transformed data spaces under the service-oriented data mining architecture. In this chapter, we propose two approaches: 1) let data providers securely agree on using a single geometric transformation; 2) securely transform different geometric perturbations to one randomly generated secret perturbation. In the following sections, we will discuss two approaches, including four protocols, in detail. Particularly, during the process of unifying data spaces, we also need to consider three problems: 1) is the privacy of any one of the data providers breached or reduced in the process? 2) if tradeoffs between the data quality and the data privacy have to be made, how to balance them? 3) what is the cost of unifying data spaces and how is it affected by other factors?

6.3 Approaches to Unifying Perturbations

Two approaches are presented in this section. For each approach we design and compare two protocols addressing different problems. The discussion will show different tradeoffs between the privacy guarantee, the data quality (model accuracy) and the cost of protocol. The first approach is ranking to find the best perturbation among the candidate perturbations. The two protocols are naive selection and threshold-based voting. The second approach is space adaptation to securely transform perturbations (data spaces) to one randomly generated perturbation. The two protocols include basic space adaptation and enhanced space adaptation with trusted party.

6.3.1 Ranking Protocols

In this section, we present the first two protocols for unifying data spaces. In both protocols, one of the local perturbation is selected as the global perturbation. We call them ranking protocols. After using either protocol to get an agreed perturbation, each data provider uses the agreed perturbation to perturb her own data and submits the perturbed data to the data miner. Since all distributed

datasets are perturbed with the same perturbation, they can be pooled together as the unified training dataset for classification modeling. Figure 94 shows the basic functional components under ranking protocols.

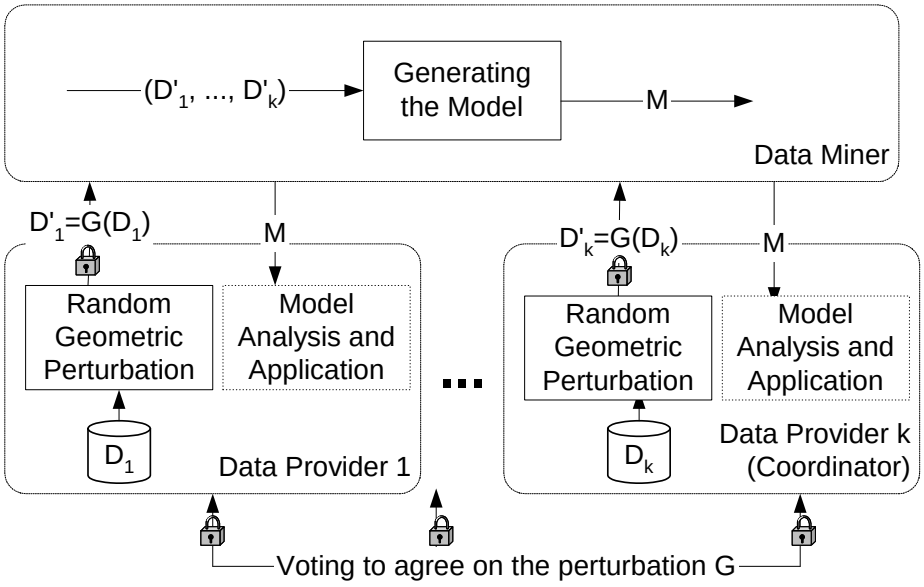

Figure 94: Unifying perturbations by ranking.

6.3.1.1 Naive Selection

The first ranking protocol is *naive selection protocol* (NS). In this protocol, each data provider generates a locally optimal perturbation and reports her own privacy guarantee to the coordinator. The coordinator simply selects the party having the highest local privacy guarantee and notifies the party to distribute her own perturbation parameters to other data providers. The following steps describe the protocol.

1. Each data provider chooses her own geometric perturbation locally with the randomized optimization algorithm [25], $G_i : (R_i, \mathbf{t_i}, \Delta_i)$, and submits the privacy guarantee q_i to the coordinator.

2. The coordinator finds the highest privacy guarantee, say p_j, and lets the data provider DP_j share her own perturbation with other data providers.

3. On receiving the DP_j's geometric perturbation G_j, each data provider submits the data perturbed by G_j to the data miner.

4. The data miner trains classification models with the collected perturbed datasets $G_j(D_i)$, $1 \leq i \leq k$ and sends the models back to the data providers.

Naive selection protocol assumes a local perturbation with high local privacy guarantee is also good for other parties. Intuitively, the global perturbation should give lower privacy guarantee than the locally optimized perturbation. This could be aggravated by "partition distribution". If we treat the sub-datasets as the partition of the pooled dataset X, how the dataset X is partitioned could affect the result. If it is a uniform partition as shown in Figure 95, the parties are likely to generate consistent perturbations, i.e. a random perturbation good for one sub-dataset will give similar privacy guarantee to other sub-datasets. Whereas, for a skewed partition, especially a class-biased partition as shown in Figure 96, the perturbations could be quite different, thus, the global perturbation will not fit every party.

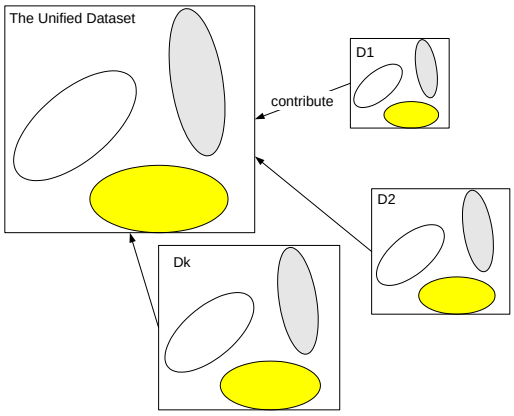

Figure 95: Uniform partition of the pooled data

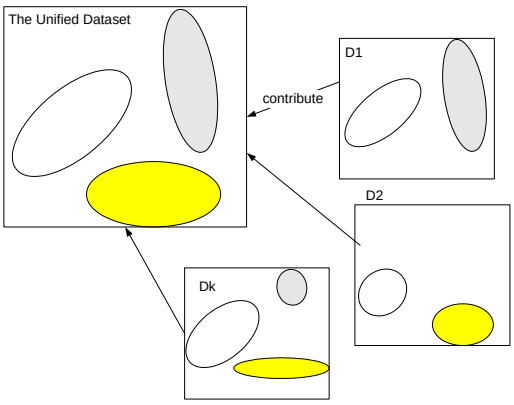

Figure 96: Class-biased partition of the pooled data

In spite of the reduction of privacy guarantee for some parties, experiments show that this simple protocol works just fine on average. We will present the next protocol, trying to reduce the sacrifice of local privacy guarantee.

6.3.1.2 *Threshold-based Voting*

The *Threshold-based voting protocol* concerns all parties' opinions to each candidate perturbation and finds one acceptable for all parties. With this protocol, each party accepts all candidate perturbations that provide privacy guarantee within the certain range of the privacy guarantee that her own perturbation can provide. This range is defined by $[(1-\theta_i)p_{ii}, p_{ii}]$, where p_{ii} is the privacy guarantee of the locally optimized perturbation and θ_i is the relaxation factor defined by the data provider DP_i.

The agreement is reached by negotiation. Each of the k data providers nominates a locally best geometric perturbation and distributes it to other $k-1$ parties. The $k-1$ candidate perturbations from others are evaluated and labeled by "accepted/rejected" according to the lower bound of privacy guarantee $(1-\theta_i)p_{ii}$. Then, the coordinator, i.e., one of the data providers, collects the voting lists, and finds an accepted candidate for all data providers. The problem is thus reduced to how to find the best candidate from the set of ranking lists, securely and efficiently.

The following steps define the process of negotiation and perturbation.

1. Each data provider chooses her own geometric perturbation with the randomized optimization algorithm described in [25], $G_i : (R_i, \mathbf{t}_i)$, and gets the privacy guarantee p_{ii}. A threshold θ_i is used to define the lower tolerance bound $(1-\theta_i)p_{ii}$. The perturbation G_i is then securely broadcast to other $k-1$ data providers.

2. On receiving the $k-1$ geometric perturbations, each data provider DP_i evaluates the privacy quality of the received perturbations G_j in terms of their own dataset. Let the results of evaluation represented by p_{ij}, $1 \leq j \leq k$, and use the lower tolerance bound $(1-\theta_i)p_{ii}$ filter out the quality values.

$$q_{ij} = \begin{cases} 1 & p_{ij} \geq (1-\theta_i)p_{ii} \\ 0 & p_{ij} < (1-\theta_i)p_{ii} \end{cases}$$

   The boolean voting vector $\{q_{i1}, q_{i2}, \ldots, q_{ik}\}$ are securely sent to the coordinator.

3. With the received voting vectors, the coordinator looks for perturbation that is acceptable to

all parties by finding the j that satisfies

$$\cap_{i=1}^{k} q_{ij} = 1$$

If it is successful, the index of the perturbation, j, is securely broadcast to the data providers. Otherwise, it initiates another round of negotiation (return to step 1).

4. Upon receiving the index of the commonly accepted perturbation G_j, each data provider sends the perturbed dataset $G_j(D_i)$ to the data miner.

5. The data miner trains classification models with the collected perturbed datasets $G_j(D_i)$, $1 \leq i \leq k$ and sends the models back to the data providers.

There are several factors affecting the termination of negotiation. First, the relaxation factor θ_i chosen by each data provider should significantly affect the efficiency and the privacy quality. The larger the θ_i is, the faster the data providers agree on a perturbation. However, this perturbation might give lower privacy quality to some parties. Second, the number of parties can also affect the efficiency. Intuitively, with increasing number of parties, the probability that all parties agree on one acceptable perturbation will decrease. Finally, the partition distribution can also affect the efficiency of negotiation and the overall privacy guarantee. We will study these factors in experiments.

6.3.1.3 Attack Analysis

In ranking protocols, since all data providers use the same perturbation known by each party, the first possible attack is "passive logging" by some curious data providers. A curious data provider can peek other data providers' communication channels and log their perturbed data when they submit the data to the data miner. Since the curious one knows the shared perturbation, the original data can be recovered from the logged data. Therefore, to prevent passive logging attacks, the perturbed data should be securely transferred to the data miner, which incurs some cost of encryption/decryption. Similar attacks can be applied to the transmitted perturbation parameters by malicious attackers. Therefore, the transferred perturbation parameters need to be encrypted too.

Second, we analyze whether the coordinator can obtain additional information to help breach other's privacy. In naive selection, the coordinator knows the local privacy guarantee for each

party's own perturbation, which leaks extra information. This can be avoided by applying a secure multiparty maximum value evaluation to find the maximum privacy guarantee [51]. In threshold-based voting, the coordinator knows only the boolean voting vectors, thus cannot perceive the exact resilience level of a particular perturbation to any individual party.

Finally, the data miner does not know the agreed perturbation – it can only estimate the original data with the perturbed data, which has been studied in single-party geometric perturbation.

6.3.1.4 Cost

Let the number of records in each data provider be n_i, $i = 1 \dots k$. Let the local perturbation optimization cost $\pi(n_i, d)$ on average [1]. If the negotiation is done in r rounds for threshold-based voting, the local optimization will cost $r\pi(n_i, d)$ on average. Naive selection costs only one round, i.e., $r = 1$. The protocol also requires the perturbation parameters to be securely transferred to each other. The cost of encryption/decrpytion for one set of parameters is proportional to the size of parameters, which is $O(d^2)$. Broadcasting the parameters to each other costs $O(k^2 d^2)$ in one round. Finally, the perturbed datasets should be encrypted too, which cost $O(d \sum n_i)$ encryption/decryption. We summarize the cost in Table 10.

Table 10: Cost for the ranking protocols.

optimization	communication	encryption/decryption
$r \sum \pi(n_i, d)$	$O(rk^2d^2 + d \sum n_i)$	$O(rkd^2 + d \sum n_i)$

6.3.2 Space Adaptation Protocols

In ranking protocols, encryption/decryption of the perturbed datasets raises a considerable extra cost. Also, the local privacy guarantee has to be sacrificed more or less to reach an agreed perturbation. In this section, we design the space adaptation protocols to address these problems. The basic space adaptation protocol does not assume the existence of the trusted third party. With the trusted party, the protocol can be further simplified and the data quality is also better preserved.

Compared to the ranking protocols, the space adaptation protocols have two unique benefits. First, it will enable the data providers to publish their perturbed datasets independently without any

[1] The cost includes the calculation of the covariance matrix and the optimization of rotation matrix, but most of the cost lies on the optimization, which is proportional to the size of the dataset (or sample datasets).

encryption. E.g., the data provider can put the dataset on web and the data miner pulls it for mining when it is needed. Second, the data provider can directly use her locally optimized perturbation, which avoids sacrificing individual privacy guarantee. However, the space adaptation protocols indeed bring some extra complexity, and possibly some reduction in model accuracy without the attending of the trusted party.

In the space adaptation protocols, each data provider needs to submit some auxiliary information − the"complementary rotation matrix" and "complementary translation vector", to the data miner, and the data miner will need to do some interactions with one of the data provider (the coordinator) to figure out the **space adaptors** for all participants. These space adaptors can transform the different perturbations into one perturbation that is randomly generated by the coordinator. To fully protect the perturbations in this protocol, we might need to sacrifice a little bit data quality in terms of classification modeling. In the following, we first give the definition and the algorithms of space adaptation, and then present the space adaptation protocols.

6.3.3 Concept of Space Adaptation

As we defined, the perturbation parameters for the data provider i are $G_i : (R_i, \mathbf{t}_i)$, the translation matrix is $\Psi_i = \mathbf{t}_i \mathbf{1}_{n_i}^t$, and the original sub-dataset is X_i. Let Y_i be the perturbed data. Now, suppose that we want to transform the perturbed dataset Y_i in the space G_i to be $Y_{i \to t}$ in the target space G_j (without the noise component). The following procedure can be applied. Since $Y_i = G_i(X_i) = R_i X_i + \Psi_i + \Delta_i$, $X_i = R_i^{-1}(Y_i - \Psi_i - \Delta_i)$. Therefore, $Y_{i \to t} = G_j(X_i) = R_j X_i + \Psi_j = R_j R_i^{-1}(Y_i - \Psi_i - \Delta_i) + \Psi_j$, and it can be further represented in the following form:

$$Y_{i \to t} = R_j R_i^{-1} Y_i - R_j R_i^{-1}(\Psi_i - \Psi_j) + (I - R_j R_i^{-1})\Psi_j - R_j R_i^{-1} \Delta_i$$

We define $R_j R_i^{-1}$ as *the rotation adaptor* R_{ij}, $\Psi_i - \Psi_j$ as *the translation adaptor* Ψ_{ij} (also represented by the corresponding translation vector $\mathbf{t}_{ij}$), and $(I - R_j R_i^{-1})\Psi_j = (I - R_{ij})\Psi_j$ as *the additional translation adaptor*, A_{ij} (also represented by the corresponding additional translation vector $\mathbf{a}_{ij}$). We will show how to securely compute the three components so that the transformation between the data spaces can be done securely.

Since Δ_i consists of iid components with $N(0, \sigma^2)$, we have $E[R_j R_i^{-1} \Delta_i] = 0$ and

$$\begin{aligned} Var[R_j R_i^{-1} \Delta_i] &= R_j R_i^{-1} Var[\Delta_i](R_j R_i^{-1})^T \\ &= R_j R_i^{-1} \sigma^2 \mathbf{I} (R_i^{-1})^T R_j^T = \sigma^2 \mathbf{I} \end{aligned}$$

i.e., the resultant noise component has the same distribution with Δ_j. Therefore, we can instead use the following equation in protocols.

$$Y_{i \to t} = R_j R_i^{-1} Y_i - R_j R_i^{-1} (\Psi_i - \Psi_j) + (I - R_j R_i^{-1}) \Psi_j \tag{28}$$

6.3.3.1 Space Adaptation Algorithm

The space adaptation algorithm consists of three components computing the three adaptors, respectively. Our algorithm enables the adaptors R_{ij} and T_{ij} to be securely calculated, while it has to make some tradeoff between the data quality and the data privacy for computing the third adaptor A_{ij}.

Securely computing the rotation adaptor The basic idea of securely computing the rotation adaptor can be summarized as follows. The data providers share a common secret key R (a randomly generated rotation matrix). With this key, each can calculate a *complementary rotation matrix* R_i' for the rotation R_i. R_i' is submitted to the data miner, who can figure out the rotation adaptor without revealing the information about the original rotation matrix.

Definition 3. *If R is any rotation matrix, the complementary matrix of R_i to R is R_i', so that $R = R_i' R_i$, i.e., $R_i' = R R_i^{-1}$*

The data miner can use only the complementary matrices to calculate the rotation adaptor. For example, the rotation adaptor for the space i to the target space j can be obtained with the complementary rotation matrices, $R_{ij} = R_j R_i^{-1} = R_j R^{-1} R R_i^{-1} = R_j'^{-1} R_i'$. As long as the common key R keeps secret, the rotation adaptor can be securely calculated.

Securely computing the translation adaptor We can easily extend the above secure-key based algorithm to securely compute the translation adaptor. Similarly, we define the *complementary translation vector* based on the secret key $\mathbf{t}$ (a randomly generated d-dimensional vector with elements in [0, 1]). Let $T = \mathbf{t}\mathbf{1}_n^t$ be the secret translation matrix.

Definition 4. *If T is any translation matrix, the complementary matrix of Ψ_i to T is Ψ'_i, so that* $\Psi'_i = T - \Psi_i$

The translation adaptor can be computed as $\Psi_{ij} = \Psi_i - \Psi_j = T - \Psi'_i - (T - \Psi'_j) = \Psi'_j - \Psi'_i$. Note that a translation matrix T is represented by the translation vector $\mathbf{t}$. We will need only the translation vectors in calculating the translation adaptor.

Securely computing the additional translation adaptor The additional translation adaptor $(I - R_{ij})\Psi_j$ involves two parts owned by two different parties, separately. The first part $I - R_{ij}$ contains the rotation adaptor R_{ij}, generated by the data miner, and Ψ_j belongs to the data provider DP_j. Both parties cannot share their parts with each other due to the two reasons: 1) if DP_j knows R_{ij}, it can infer R_i and try it to all publicly available perturbed datasets. Potentially, some of the data provider DP_i's privacy can be breached. 2) if the data miner knows Ψ_j, it can infer any other T_i through the known T_{ij}s, which causes the privacy breach to the points around the rotation center [24].

In order to securely calculate $(I - R_{ij})\Psi_j$ (in practice, $(I - R_{ij})\mathbf{t}_j$), we design the following algorithm.

1. The data miner randomly generates a rotation matrix R_r as the protection matrix and sends $R_r(I - R_{ij})$ to DP_j.

2. DP_j generates a random vector $\mathbf{r}_i$ and returns $R_r(I - R_{ij})\mathbf{t}_j + \mathbf{r}_i$ to the data miner.

3. The data miner calculates the approximate result by $R_r^{-1}(R_r(I-R_{ij})\mathbf{t}_j+\mathbf{r}_i)$ = $(I-R_{ij})\mathbf{t}_j+R_r^{-1}\mathbf{r}_i$, which contains the error component $R_r^{-1}\mathbf{r}_i$ (type (1) error).

Proposition 13. *It is probabilistically difficult to breach R_{ij} and $\mathbf{t}_j$ with the above protocol.*

SKETCH PROOF. DP_j cannot effectively infer R_{ij} from $R_r(I - R_{ij})$ due to the unknown component R_r. Let $(I - R_{ij})\mathbf{t}_j + R_r^{-1}\mathbf{r}_i = \hat{\mathbf{t}}_{ij}$. If the data miner wants to estimate $\mathbf{t}_j$ with $\hat{\mathbf{t}}_j = (I-R_{ij})^{-1}\mathbf{t}_{ij} = \mathbf{t}_j+(I-R_{ij})^{-1}R_r^{-1}\mathbf{r}_i$, it will result in an error component $(I-R_{ij})^{-1}R_r^{-1}\mathbf{r}_i$. We name it type (2) error.

One potential way to estimate $\mathbf{t}_j$ is to formally figure out the distribution of the type (2) error , so that the estimate of $\mathbf{t}_j$ has certain statistical guarantee. However, it cannot be achieved due to

the combined components. Let $A = (I - R_{ij})^{-1}$, $B = R_r^{-1}$, $C = \mathbf{r}_i$ make up the type (2) error. Although B can be fixed, the distributions of A and B are unknown. Therefore, there is no robust estimation for the combination of the three components. □

Type (1) error reduces the data quality and may cause the reduction of model accuracy, while type (2) error protects $\mathbf{t}_j$ from attacks. Ideally, we want small type (1) error so that our model accuracy is preserved, and we also prefer larger type (2) error to preserve privacy better. Since type (2) error is type (1) error multiplied by the factor $(I - R_{ij})^{-1}$, we expect small type (1) error will introduce considerably large type (2) error. We show in experiments how the type (1) and (2) error components are related to the noise component $\mathbf{r}$, and we will also experimentally study the effect of noise addition to the model accuracy.

6.3.3.2 The Basic Space Adaptation Protocol

Figure 97 shows the components and interactions in the basic space adaptation protocol. With the

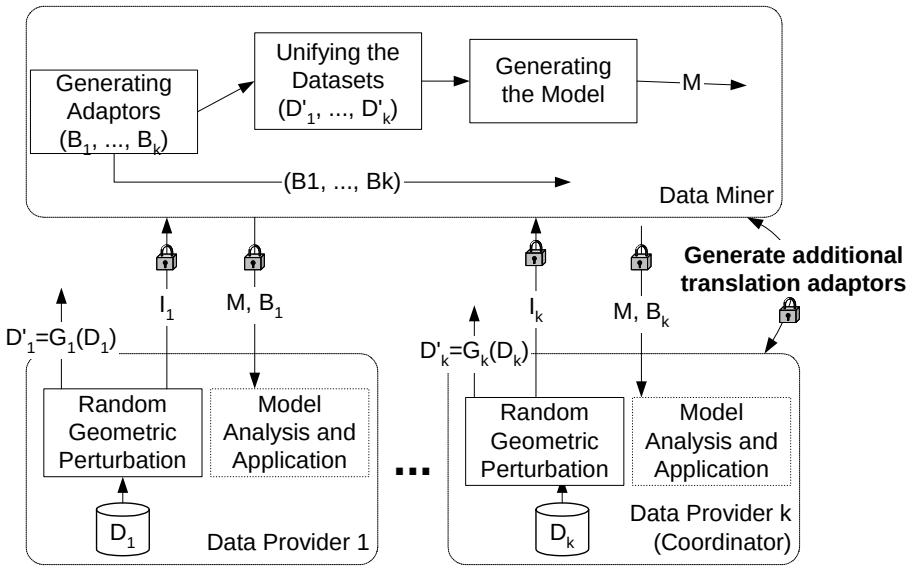

Figure 97: Space adaptation.

space adaptation algorithms, the protocol can be described as follows.

1. One of the data providers, without loss of generality, DP_1, initiates a randomly generated rotation matrix R and a randomly generated translation vector $\mathbf{t}$, and then securely propagates R and $\mathbf{t}$ to other data providers;

2. Each data provider DP_i uses the local perturbation optimization algorithm to find a locally optimized perturbation $G_i : (R_i, \mathbf{t}_i)$. It also calculates the complementary rotation matrix and complementary translation vector with the received R and $\mathbf{t}$. Let the auxiliary information be $I_i = (R'_i, \mathbf{t}'_i)$. $G_i(X_i)$ is published and I_i is securely sent to the data miner;

3. Upon receiving the perturbed data and the auxiliary information, the data miner randomly selects a data provider, WLOG, DP_k as the coordinator. DP_k also generates a *virtual data space* $G_v : (R_v, \mathbf{t}_v)$ as the target space and submits the auxiliary information of G_v to the data miner.

4. With the auxiliary information, the data miner is able to calculate the rotation adaptors and the translation adaptors from G_i to G_v, respectively. With the help DP_k, the data miner also approximately calculates the additional translation adaptors.

5. With the space adaptors, the data miner transforms the perturbed datasets into the space G_v by using the formula (28). Then, it trains the classification models and sends them back securely with the adaptors $(R_{iv}, \mathbf{t}_{iv}, \mathbf{a}_{iv})$, so that the data provider can transform their perturbed data to the target space with the adaptors in order to use the classification models.

6.3.3.3 The Space Adaptation Protocol with the Trusted Third Party

In the basic space adaptation protocol, the secure calculation of additional translation adaptor requires noise addition, which may result in reduction of model accuracy. In this section, we show, when a trusted third party exists, the protocol can be simpler and there is no additional tradeoff between the privacy guarantee and the model accuracy. Figure 98 also shows the interactions between the parties.

Basically, the trusted party handles all space adaptation issues. The data providers do not need to provide the auxiliary information any more. They simply submit their perturbation parameters to the trusted party. The trusted party randomly generates a target space and calculates the three kind of adaptors to the target space for each perturbation. the data miner then obtains the adaptors from the trusted party and returns the mined models to the data providers. Concretely, the protocol has the following steps.

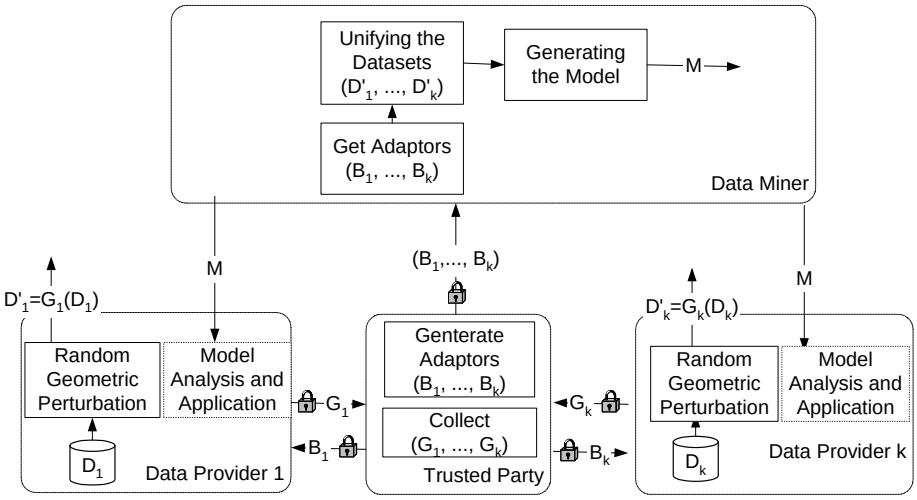

Figure 98: Space adaptation with a trusted party.

1. Each data provider DP_i uses the local perturbation optimization algorithm to find a locally optimized perturbation $G_i : (R_i, \mathbf{t}_i)$. $G_i(X_i)$ are published and $(R_i, \mathbf{t}_i)$ is securely sent to the trusted party.

2. Upon receiving the perturbation parameters from all data providers, the trusted party randomly generates a target space $G_v : (R_v, \mathbf{t}_v)$, and calculates the adaptors $(R_{iv}, \mathbf{t}_{iv}, \mathbf{a}_{iv})$ to the target space for each data provider.

3. The data miner requires the adaptors for each data provider DP_i and transforms $G_i(X_i)$ to G_v. Then, it trains the classification models and sends them to the data providers.

4. Each data provider securely gets the adaptors from the trusted party, and transforms their data to the target space in order to use the classification models.

6.3.3.4 Attack Analysis

First of all, the data miner cannot discover any original data or the secret perturbation parameters from the auxiliary information it gets. Since the data miner does not know the secret matrix R and the secret vector $\mathbf{t}$, it cannot infer the original perturbation parameters. As we discussed in section 6.3.3.1, neither can the data miner effectively estimate the translation vector of the target space, $\mathbf{t}_v$, in calculating the additional translation adaptors.

Second, we consider the possible attacks from the data providers. 1) One singular scenario

could possibly happen that some of the data providers have their own perturbation parameters, (i.e., the rotation matrix and translation vector), close to the shared secret rotation matrix R and the secret translation vector $\mathbf{t}$. If two sub-datasets have the same distribution, one could infer the other's perturbation from her own perturbation. Therefore, the data providers should avoid to generate parameters close to the shared secrets. 2) If a data provider knows others' auxiliary information, s/he can infer their original perturbation, too. Therefore, the auxiliary information should be securely transferred to the data miner. 3) The data providers also know the target space – they can simply figure out the target space $G_v : (R_v, \mathbf{t}_v)$ based on the adaptors received from the data miner. However, knowing the target space does not help breach other data provider's privacy, since there is no published perturbed data in the target space.

Third, the coordinator gets no additional information from the interaction with the data miner as we discussed in section 6.3.3.1.

6.3.3.5 Cost

The data providers need to do local optimization once. Securely distributing R and $\mathbf{t}$ requires $O(kd^2)$ encryption/decryption and $O(kd^2)$ communication cost. Securely transferring the auxiliary information to the data miner also requires $O(kd^2)$ communication cost and $O(kd^2)$ encryption/decryption cost. The additional communication cost in the interaction between the data miner and the coordinator (or the trusted party) is also $O(kd^2)$. Finally, the data miner sends back the adaptors and models, which again costs $O(kd^2)$ in communication and encryption/decryption, respectively. We summarize the total cost in Table 11. Compared to ranking protocols, the cost of space adaptation is significantly reduced in all of the three aspects.

Table 11: Cost for space adaptation protocols.

optimization	communication	encryption/decryption
$\sum \pi(n_i, d)$	$O(kd^2 + d\sum n_i)$	$O(kd^2)$

6.4 Experiments

In this section, we present two sets of experiments, exploring those issues we mentioned in the early sections. 1) For the ranking protocols, we will first study the effectiveness of naive selection, and then explore the relationship between the efficiency of negotiation and the potential factors, such

as the partition of the sub-datasets and the relaxation factor. 2) For the space adaptation protocol (without the trusted party attending), we will study the potential tradeoffs between the data quality (represented by the model accuracy) and the data privacy, which happens in calculating the additional translation adaptors. With the trusted party, the tradeoff does not happen. Before starting the discussion, we give the basic setting of the experiments first.

6.4.1 Setting of Experiments

The local perturbation optimization algorithm integrates the fastICA implementation[2] to test the resilience of the candidate perturbation to the ICA-based attacks (and their enhancement with the knowledge of data distribution) [24]. In the classifier testing, we use two representative classifiers: KNN classifer and SVM with radial basis function as the kernel. The SVM implementation is from LIBSVM[3]. In our KNN implementation, we also use the kd-tree implemented in ANN library[4] to efficiently search the nearest neighbors. Thanks to the authors of these great tools.

We use 12 UCI datasets in the experiments. Due to the relatively small data size, each dataset is split into several randomly sized sub-datasets, simulating the distributed datasets from the data providers. We also simulate the two special partition methods: the class-biased partition and the uniform partition, as shown in Figure 95 and 96. The minimum privacy guarantee (Section 6.2.1) is used in the evaluation of privacy guarantee for an individual party.

In most experiments, we also choose to show the detailed results of three typical datasets: diabetes dataset that has an unclear geometric class boundary (KNN with accuracy about 73%), iris dataset that has geometrically well-separated three classes (KNN with accuracy about 95%), and votes dataset that is a boolean dataset. θ_i is set to the same θ level for each party.

6.4.2 Results of Naive Selection

In naive selection protocol, the perturbation having the highest local privacy guarantee is selected as the global perturbation for all parties. However, this perturbation may not be better than the one locally optimized by the individual party. Figure 99 shows the results of 10 runs on the three datasets

[2]http://www.cis.hut.fi/projects/ica/fastica/
[3]http://www.csie.ntu.edu.tw/ cjlin/libsvm/
[4]http://www.cs.umd.edu/ mount/ANN/

with the two partition distributions, respectively. Let G_i be the selected perturbation, and p_{ij} be the privacy guarantee applied to the party j. "Selected local maximum" is the selected perturbation that has the maximum local privacy guarantee, i.e., p_{ii}. "Maximum Applied" $= max\{p_{ij}, j = 1 \ldots k\}$, "Minimum Applied"$= min\{p_{ij}, j = 1 \ldots k\}$, and "Average Applied"$= 1/k \sum_{j=1}^{k} p_{ij}$, summarize the privacy guarantees of all parties when the selected perturbation is applied to other parties.

The "Average Applied" is significantly lower than the "Selected local maximum", which means the selected perturbation cannot give the same privacy guarantee to all parties as to the selected party. Although the "Average Applied" value is not low, some parties (the "Minimum Applied") may have much lower privacy guarantee than the average value. Overall, uniform partition gives less variance over different parties – the gap between the Maximum Applied and Minimum Applied is smaller.

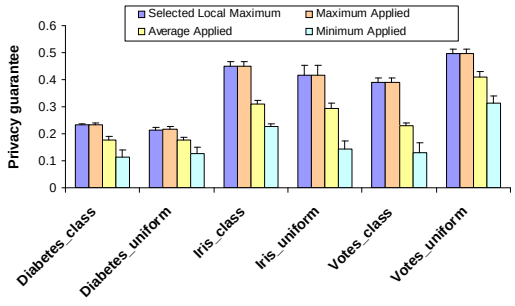

Figure 99: All parties' privacy guarantee with naive selection (the number of parties is 6) .

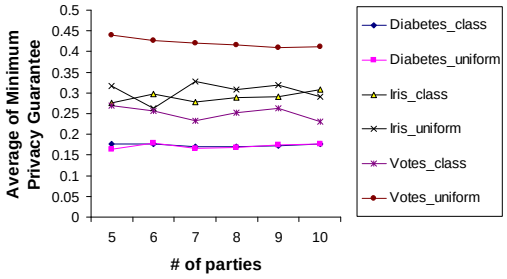

Figure 100: The effect of the number of parties to the average privacy guarantee of all parties.

We also studied the effect of the number of parties to the average privacy guarantee of all parties. Figure 100 shows the two factors are not quite correlated. Therefore, although naive selection may result in lower privacy guarantee for some parties, it should be scalable to the number of parties.

6.4.3 Negotiation in Threshold-based Voting

In this set of experiments, we evaluate the effect of multiple factors to the threshold-based voting protocol. We run 100 negotiations for each dataset and each partition mode. Figure 101 and 102 show the number of successful negotiations in the 100 runs for each θ setting. Comparing the two Figures, we find that different partition distributions indeed significantly affect the efficiency of negotiation. The success rate of uniform partition is much higher than that of class-biased partition for

the same θ setting. Different datasets (which have different class distributions) also have significant impact on the efficiency of negotiation for class-biased partition.

Figure 103 and 104 present the gain of privacy guarantee (compared to the "Minimum Applied" in Figure 99) by using threshold-based voting instead of naive selection. Uniform partition benefits more from the negotiation, and the gains may vary from dataset to dataset. When the θ becomes sufficiently large the gain may become negative for some dataset/partition (as Iris_class at $\theta > 0.1$), i.e., the negotiation-based protocol becomes even worse than naive selection, but for most datasets/partitions, the gain is significant.

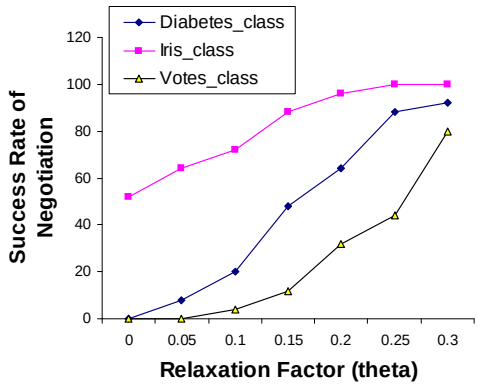

Figure 101: Success rate of negotiation with class-biased partition.

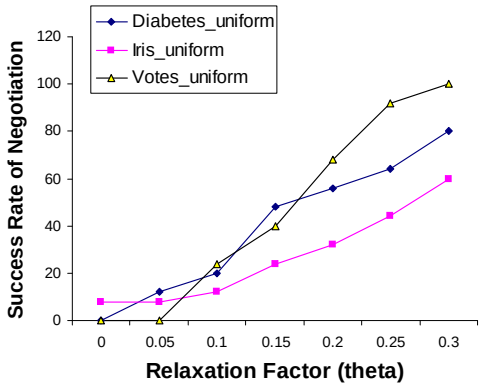

Figure 102: Success rate of negotiation with uniform partition.

However, different from naive selection, the efficiency of negotiation is quite limited by the number of parties. With the increasing number of parties, the probability that all parties agree on one perturbation may be decreasing. With $\theta = 0.15$, experimental result (Figure 106) shows that

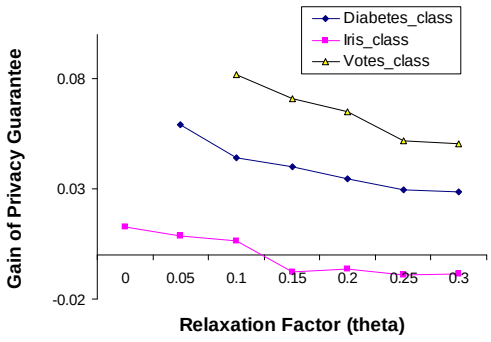

Figure 103: Gain of privacy guarantee in negotiation with class-biased partition.

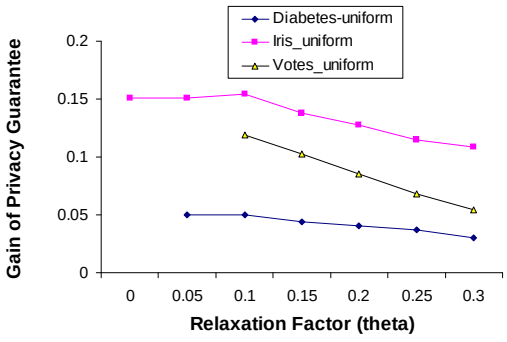

Figure 104: Gain of privacy guarantee in negotiation with uniform partition.

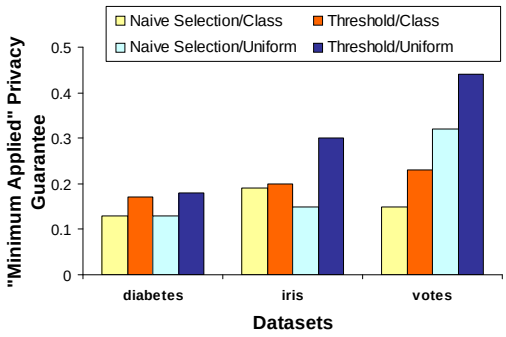

Figure 105: Privacy guarantee with different partition modes

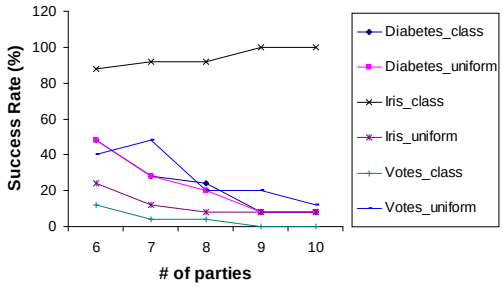

Figure 106: The success rate of negotiation vs. the number of parties for different dataset/partition-distribution. ($\theta = 0.1$)

the success rate drops significantly in most cases as the number of parties increases from 5 to 10. Due to the small size and special class distribution, the success rate of Iris_class keeps exceptionally high. This indicates the scalability of the threshold-based protocol is not so good.

6.4.4 Tradeoffs in Basic Space Adaptation Protocol

In the threshold-based voting protocol, the tradeoff is primarily made on the efficiency of negotiation and the privacy guarantee, while the data quality is not affected since all parties use the same perturbation. In the space adaptation protocol, we need to add in some error components when we calculate the additional translation adaptor, which can possibly results in the reduction of data quality. In this set of experiments, we show how the errors are related and how they affect the data quality (and the model accuracy).

We are interested in the type (1) and type (2) errors defined earlier. As we discussed, type (1) error causes reduction of data quality, while type (2) error preserves privacy. The two are related by the factor $(I - R_{ij})^{-1}$. We expect that this factor $(I - R_{ij})^{-1}$ makes error (2) much larger than error (1), ideally matching our goal: introducing smaller error (1) to the dataset while maximizing the difficulty (the error (2)) in estimating the translation vector.

In experiments, the coordinator's noise component $\mathbf{r}_i$ is generated with normal distribution $N(0, \sigma^2)$, and we tune the variance σ^2 to change the intensity of the noise component. A fixed rotation matrix R_r is used as the data miner's protection matrix. Similarly, we also study the effect of different partition distributions to the results. Figure 107 and 108 show the change of error

(1) and (2) with the varying variance for two types of partition distributions, respectively. "Diabetes_Error(1)" means the error (1) of the dataset Diabetes. Error (2) is usually times larger than error (1). We can also see error (1) is quite stable for different datasets since the same two components R_r and $\mathbf{r}_i$ are used, while error (2) can vary unpredictably due to the factor $(I - R_{ij})^{-1}$.

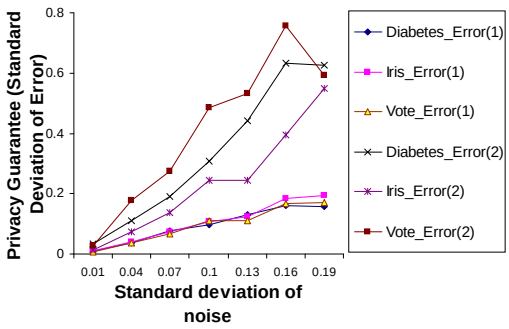

Figure 107: Noise vs. errors, uniform partition

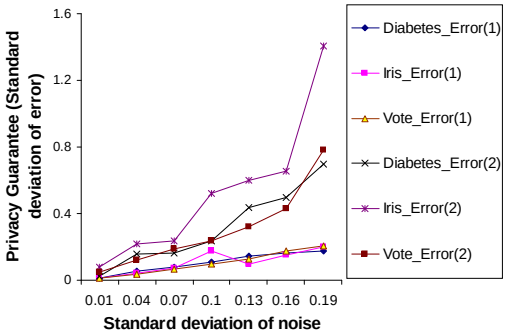

Figure 108: Noise vs. errors, class-biased partition

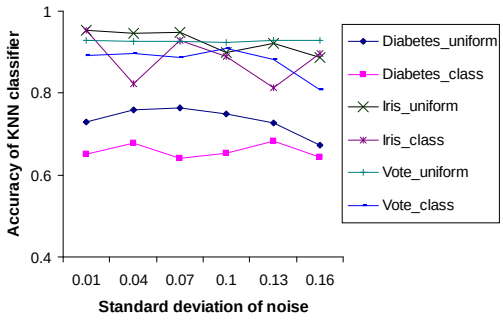

Figure 109: Noise addition vs. model accuracy for KNN

On the other hand, we also need to consider the effect of noise addition to the model accuracy. Figure 109 is the result of KNN classifier, showing that the effect of partition distribution can be significant. For some dataset with certain partition method, such as iris with class-biased partition, the accuracy can fluctuate in large range. Overall, the accuracy tends to slightly decrease with the increase of the noise variance.

We finalize the discussion with more results for the two representative classifiers, KNN classifier and SVM classifier with RBF kernel. The numbers in Figure 110 and 111 show the deviation from the standard accuracy with the standard deviation of noise level at 0.1. A negative number means that the actual accuracy is reduced. It shows that KNN classifier is more sensitive to the noise

addition, while there is little effect on SVM classifier. Also, quite surprisingly, class-biased partition often increases the accuracy of SVM classifier.

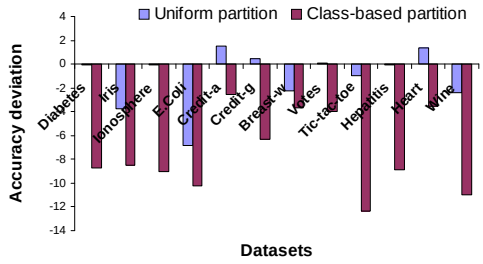

Figure 110: The average deviation of model accuracy for KNN classifier.

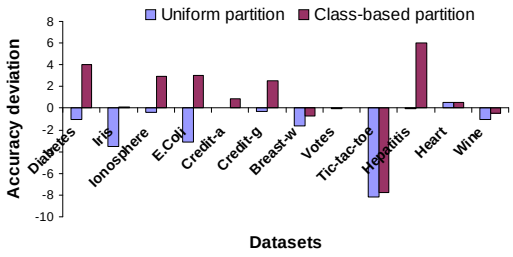

Figure 111: The average deviation of model accuracy for SVM(RBF) classifier.

In summary, we studied two approaches: the ranking approach and the space adaptation approach. In the ranking approach, naive selection protocol tends to reduce some parties' privacy guarantee, while threshold-based voting protocol allows each party to specify an acceptable range of privacy guarantee and tries to reach an perturbation giving satisfactory privacy guarantee to all parties. However, threshold-based voting is not so scalable to the number of parties. The ranking protocols require the perturbed data to be transmitted securely and some parties may have to use a perturbation with not high privacy guarantee. The space adaptation approach addresses these problems and allows each party to use their own perturbation. However, the space adaptation approach involves some tradeoff between privacy guarantee and data quality if the trusted party is not present. We also experimentally studied these protocols and the related factors and get some initial results on performing appropriate tradeoffs to get good balance between the privacy guarantee, the model accuracy and the efficiency of unifying perturbations.

CHAPTER VII

CONCLUSION

This thesis consists of three major components, describing three novel applications of geometric methods in mining large and possibly private datasets, respectively. The first component is the iVIBRATE interactive-visualization based three-phase framework for clustering large datasets. Its VISTA subsystem allows users to interactively view the potential clusters in continuously changing visualizations, particularly effective for finding irregularly shaped clusters. More importantly, it can incorporate, validate, and refine the results of any automated clustering algorithms. Its adaptive ClusterMap labeling subsystem preserves the irregular cluster boundary defined in VISTA subsystem, clearly distinguishes the outliers and provides effective mechanisms for fine-tuning and flexible refinement of boundary extensions during the disk labeling phase. We also thoroughly discussed the issues and solutions in integrating the three phases under the iVIBRATE framework, aiming at providing a reliable and stable framework. Experimental results show that the iVIBRATE framework can effectively reduce error rates caused by irregularly shaped clusters, domain-specific definition, outliers, and rough disk-labeling. Most importantly, it allows the user to monitor where the main errors occur in the entire process.

The second component is the Best K method for determining the best K number of clusters in clustering categorical data. Specifically, we address three problems: 1) identifying the best Ks for categorical data clustering; 2) determining whether a dataset contains significant clustering structure; 3) developing theory for handling large datasets. Our idea is to find the best Ks by observing the entropy difference between the neighboring clustering results of K and $K + 1$ clusters, respectively. The "Best-K plot (BKPlot)" is used to conveniently identify the critical Ks. We also develop an entropy-based hierarchical algorithm ACE, which is the most robust method to generate high-quality approximate BKPlots. Furthermore, the theory of sample BKPlots is developed for finding the best Ks for very large datasets. The result is extended to analyzing the datasets having no clustering structure. Based on the features of sample no-cluster datasets, a statistical test is developed

for determining whether a given dataset has significant clustering structure. Experimental results support that the BKPlot method is effective in determining the best K number of clusters for categorical clustering, and the ACE algorithm is the most effective algorithm in generating high-quality approximate BKPlots among the existing algorithms.

The third component is a random geometric perturbation approach to privacy preserving data classification. Random geometric perturbation, $G(X) = RX + \Psi + \Delta$, includes the linear combination of the three components: random rotation, random translation, and random noise addition. Geometric perturbation can preserve the important geometric properties, thus the model accuracy of most classifiers that search for geometric class boundaries is well preserved with the perturbed data. We proved that the three popular types of classifiers (kernel methods, SVM classifiers with certain kernels, and hyperplane-based classifiers) are all invariant to rotation perturbation. The distance-based classifiers are also invariant to translation perturbation. A new multi-column privacy model is proposed to analyze the privacy property of geometric perturbation. Three kinds of inference attacks: naive-inference, ICA-inference, and distance-inference, are studied to optimize the privacy guarantee for a given dataset. Experimental results show that the proposed randomized geometric perturbation optimization can effectively find perturbations of high privacy guarantee for most experimental datasets, and geometric perturbation can provide higher privacy guarantee than the existing multidimensional perturbation method, while well preserving model accuracy.

We also studied the application of geometric perturbation to service-oriented multiparty collaborative privacy-preserving data classification. The main challenge is to unify the different perturbations used by the collaborative data providers. Since each perturbation can be regarded as a transformed data space, we aim at unifying all perturbed datasets into one data space. For this purpose, we propose three protocols: the voting protocol, the space adaptation protocol, and the space adaptation protocol with trusted party. We analyzed the features and the cost of each protocol, and also studied the relationship between the main factors and tradeoffs in experiments. Experiments show that the protocols can be possibly implemented with a good balance between the privacy guarantee, the model accuracy, and the computational cost.

To summarize, this thesis explores the geometric properties of the multidimensional datasets in statistical learning and data mining, and provides novel techniques and frameworks for mining large

(and distributed) datasets while protecting the desired data privacy. The examples in this thesis have shown that geometric methods are among the most convenient tools for investigating the properties of multidimensional datasets.

This thesis has well addressed several challenges in the three projects. However, there are also some open research issues worth future efforts.

1. We use the interactive visualization framework to address the problems in clustering large numerical datasets that automatic approaches cannot effectively handle. However, visualization has its own challenges, mostly in visualizing multidimensional datasets. Experiments show that VISTA visualization can find better separation of clusters than automatic algorithms in most cases. However, we guess that VISTA visualization model cannot visualize some special cluster distributions $-$ there is no VISTA visualization that can well separate clusters. We have proposed to use multiple visualizations to separate different pairs of clusters. How to theoretically characterize the features of clustering structures that VISTA model can (or cannot) successfully visualize is a challenging problem. Randomized rendering is one way to address the efficiency problem of visual rendering. We believe theoretical analysis of clustering structures could also help to develop new method to improve the efficiency of rendering. VISTA visual rendering has shown the exceptional ability in analyzing clusters in experiments. We also believe its application will be fruitful.

2. The BKPlot method has provided the first batch of results for the best K problem in categorical data clustering. The approximate BKPlots generated by ACE algorithm still have some false discovery of best-K candidates for some datasets. Particularly, the false discovery happens more likely at the smallest K, $K = 2$. We guess this could be the result of the agglomerative nature of ACE algorithm. In addition, identifying the best K for the evolving clustering structures in data streams is another challenging problem. We have developed some initial results [26], however, the problems appear much more complicated than those in static (large) datasets. The last issue is related to special (or domain-specific) categorical data clustering, such as transactional data clustering. The BKPlot method can provide some candidates of best-K in terms of generic entropy-based clustering criterion. The candidates normally are

the best Ks in different level of clustering structure. To accept them all or select one or some may depend on the specific domain-specific measures.

3. In the basic theory of geometric perturbation, we have identified several inference attacks in terms of the different knowledge level of attackers. It is critical to study more possible attacks so that the geometric perturbation can be reinforced. The second issue is about privacy evaluation. The current variance-based multi-column privacy evaluation model is one of the most effective and convenient methods to evaluate the privacy guarantee. However, it is possible that other (multi-column) privacy evaluation models may work better for evaluating the effectiveness of some particular attacks, which might be discovered in the future. For multiparty collaborative data classification with geometric perturbation, we have proposed two approaches and four protocols. Each one has advantages and limitations. Further study on the features of these protocols would result in better understanding and the development of new protocols. In addition, geometric perturbation is currently designed for data classification problem. It is also valuable to extend the study to other data mining models.

We believe that these open issues are all interesting and challenging, and the geometric properties of data mining models will continue to play an important role in studying these open issues.

REFERENCES

[1] AGGARWAL, C. C., MAGDALENA, C., and YU, P. S., "Finding localized associations in market basket data," *IEEE Trans. on Knowledge and Data Eng.*, vol. 14, no. 1, pp. 51–62, 2002.

[2] AGGARWAL, C. C. and YU, P. S., "A condensation approach to privacy preserving data mining," *Proc. of Intl. Conf. on Extending Database Technology (EDBT)*, vol. 2992, pp. 183–199, 2004.

[3] AGRAWAL, D. and AGGARWAL, C. C., "On the design and quantification of privacy preserving data mining algorithms," *Proc. of ACM PODS Conference*, 2002.

[4] AGRAWAL, R. and SRIKANT, R., "Privacy-preserving data mining," *Proc. of ACM SIGMOD Conference*, 2000.

[5] AGRAWAL, S. and HARITSA, J. R., "A framework for high-accuracy privacy-preserving mining.," *Proc. of IEEE Intl. Conf. on Data Eng. (ICDE)*, pp. 193–204, 2005.

[6] AGRESTI, A., *Categorical Data Analysis*. Wiley-Interscience, 1990.

[7] ANDRITSOS, P., TSAPARAS, P., MILLER, R. J., and SEVCIK, K. C., "Limbo:scalable clustering of categorical data," *Proc. of Intl. Conf. on Extending Database Technology (EDBT)*, 2004.

[8] ANKERST, M., BREUNIG, M. M., KRIEGEL, H.-P., and SANDER, J., "OPTICS: Ordering points to identify the clustering structure," *Proc. of ACM SIGMOD Conference*, pp. 49–60, 1999.

[9] ASIMOV, D., "The grand tour: A tool for viewing multidimensional," *SIAM Journal of Scientific and Statistical Computing*, vol. 6, no. 1, pp. 128–143, 1985.

[10] BAEZA-YATES, R. and RIBEIRO-NETO, B., *Modern Information Retrieval.* New York City, NY: Addison Wesley, 1999.

[11] BARBARA, D. and JAJODIA, S., eds., *Applications of Data Mining in Computer Security*. Klumer Academic Publishers, 2002.

[12] BARBARA, D., LI, Y., and COUTO, J., "Coolcat: an entropy-based algorithm for categorical clustering," *Proc. of ACM Conf. on Information and Knowledge Mgt. (CIKM)*, 2002.

[13] BAULIEU, F., "Two variant axiom systems for presence/absence based dissimilarity coefficients," *Journal of Classification*, vol. 14, 1997.

[14] BAXEVANIS, A. and OUELLETTE, F., eds., *Bioinformatics: A Practical Guide to the Analysis of Genes and Proteins, 2nd edition*. Wiley-Interscience, 2001.

[15] BOCK, H., "Probabilistic aspects in cluster analysis," *Conceptual and Numerical Analysis of Data*, 1989.

[16] BRADLEY, P. S., FAYYAD, U. M., and REINA, C., "Scaling clustering algorithms to large databases," *Proc. of ACM SIGKDD Conference*, pp. 9–15, 1998.

[17] BRAND, M., "An entropic estimator for structure discovery," in *Proc. Of Neural Information Processing Systems (NIPS)*, pp. 723–729, 1998.

[18] BUJA, A., COOK, D., and SWAYNE, D. F., "Interactive high-dimensional data visualization," *Journal of Computational and Graphics Statistics*, vol. 5, pp. 78–99, 1996.

[19] CELEUX, G. and GOVAERT, G., "Clustering criteria for discrete data and latent class models," *Journal of Classification*, 1991.

[20] CHAKRABARTI, D., PAPADIMITRIOU, S., MODHA, D. S., and FALOUTSOS, C., "Fully automatic cross-associations," *Proc. of ACM SIGKDD Conference*, 2004.

[21] CHEN, K. and LIU, L., "Clustermap: Labeling clusters in large datasets via visualization," *Proc. of ACM Conf. on Information and Knowledge Mgt. (CIKM)*, pp. 285–293, 2004.

[22] CHEN, K. and LIU, L., "VISTA: Validating and refining clusters via visualization," *Information Visualization*, vol. 3, no. 4, pp. 257–270, 2004.

[23] CHEN, K. and LIU, L., "The "best k" for entropy-based categorical clustering," *Proc. of Intl. Conf. on Scientific and Statistical Database Management (SSDBM)*, pp. 253–262, 2005.

[24] CHEN, K. and LIU, L., "A random geometric perturbation approach to privacy-preserving data classification," *Technical Report*, 2005.

[25] CHEN, K. and LIU, L., "A random rotation perturbation approach to privacy preserving data classification," *Proc. of Intl. Conf. on Data Mining (ICDM)*, 2005.

[26] CHEN, K. and LIU, L., "Detecting the change of clustering structure in categorical data streams," *SIAM Data Mining Conference*, 2006.

[27] CHENG, C. H., FU, A. W.-C., and ZHANG, Y., "Entropy-based subspace clustering for mining numerical data," *Proc. of ACM SIGKDD Conference*, 1999.

[28] CLIFTON, C., "Tutorial: Privacy-preserving data mining," *Proc. of ACM SIGKDD Conference*, 2003.

[29] CONOVER, W., *Practical Nonparametric Statistics*. New York City, NY: John Wiley and Sons, 1998.

[30] COOK, D., BUJA, A., CABRERA, J., and HURLEY, C., "Grand tour and projection pursuit," *Journal of Computational and Graphical Statistics*, vol. 23, pp. 155–172, 1995.

[31] COVER, T. and THOMAS, J., *Elements of Information Theory*. Wiley, 1991.

[32] COX, T. F. and COX, M. A. A., *Multidimensional Scaling*. Boca Raton, FL, US: Chapman&Hall/CRC, 2001.

[33] CRISTIANINI, N. and SHAWE-TAYLOR, J., *An Introduction to Support Vector Machines and Other Kernel-based Learning Methods*. Cambridge University Press, 2000.

[34] CUTTING, D. R., KARGER, D. R., PEDERSEN, J. O., and TUKEY, J. W., "Scater/gather: a cluster-based approach to browsing large document collections," *Proc. of ACM SIGIR Conference*, pp. 318–329, 1992.

[35] DHILLON, I. S., MELLELA, S., and MODHA, D. S., "Information-theoretic co-clustering," *Proc. of ACM SIGKDD Conference*, 2003.

[36] DHILLON, I. S., MODHA, D. S., and SPANGLER, W. S., "Visualizing class structure of multidimensional data," *the 30th Symposium on the Interface: Computing Science and Statistics*, vol. 30, pp. 488–493, 1998.

[37] DUBES, R. C. and JAIN, A. K., "Validity studies in clustering methodologies," *Pattern Recognition*, pp. 235–254, 1979.

[38] (ED.), C. B., *Data Visualization Techniques (Trends in Software 6)*. John Wiley & Sons, 1999.

[39] ESTER, M., KRIEGEL, H.-P., SANDER, J., and XU, X., "A density-based algorithm for discovering clusters in large spatial databases with noise," *Second International Conference on Knowledge Discovery and Data Mining*, pp. 226–231, 1996.

[40] EVFIMIEVSKI, A., GEHRKE, J., and SRIKANT, R., "Limiting privacy breaches in privacy preserving data mining," *Proc. of ACM PODS Conference*, 2003.

[41] EVFIMIEVSKI, A., SRIKANT, R., AGRAWAL, R., and GEHRKE, J., "Privacy preserving mining of association rules," *Proc. of ACM SIGKDD Conference*, 2002.

[42] FALOUTSOS, C. and LIN, K.-I. D., "FastMap: A fast algorithm for indexing, data-mining and visualization of traditional and multimedia datasets," *Proc. of ACM SIGMOD Conference*, pp. 163–174, 1995.

[43] FEIGENBAUM, J., ISHAI, Y., MALKIN, T., NISSIM, K., STRAUSS, M., and WRIGHT, R. N., "Secure multiparty computation of approximations," in *ICALP '01: Proceedings of the 28th International Colloquium on Automata, Languages and Programming,*, pp. 927–938, Springer-Verlag, 2001.

[44] FEINGOLD, CORZINE, WYDEN, and NELSON, "Data-mining moratorium act of 2003," *U.S. Senate Bill (proposed)*, January 16 2003.

[45] FREIDMAN, J. H., BENTLEY, J. L., and FINKEL, R. A., "An algorithm for finding best matches in logarithmic expected time," *ACM Trans. on Mathematical Software*, vol. 3, no. 3, pp. 209–226, 1977.

[46] FUNG, B. C., WANG, K., and YU, P. S., "Top-down specialization for information and privacy preservation," *Proc. of IEEE Intl. Conf. on Data Eng. (ICDE)*, 2005.

[47] GALLIER, J., *Geometric Methods and Applications for Computer Science and Engineering*. New York: Springer-Verlag, 2000.

[48] GANTI, V., GEHRKE, J., and RAMAKRISHNAN, R., "CACTUS-clustering categorical data using summaries," *Proc. of ACM SIGKDD Conference*, 1999.

[49] GIBSON, D., KLEINBERG, J., and RAGHAVAN, P., "Clustering categorical data: An approach based on dynamical systems," *Proc. of Very Large Databases Conference (VLDB)*, vol. 8, no. 3–4, pp. 222–236, 2000.

[50] GOLDREICH, O., *Foundations of Cryptography*. Cambridge University Press, 2001.

[51] GOLDREICH, O., "Secure multiparti computation," *The Foundation of Cryptography*, vol. 2, 2004.

[52] GRAY, J., "What next? a few remaining problems in information technlogy, sigmod conference 1999, acm turing award lecture, video," *ACM SIGMOD Digital Symposium Collection*, vol. 2, no. 2, 2000.

[53] GUHA, S., RASTOGI, R., and SHIM, K., "CURE: An efficient clustering algorithm for large databases," *Proc. of ACM SIGMOD Conference*, pp. 73–84, 1998.

[54] GUHA, S., RASTOGI, R., and SHIM, K., "ROCK: A robust clustering algorithm for categorical attributes," *Information Systems*, vol. 25, no. 5, pp. 345–366, 2000.

[55] GUO, S. and WU, X., "Deriving private information from general linear transformation perturbed data," *CS Technical Report, UNC Charlotte*, March, 2006.

[56] HALKIDI, M., BATISTAKIS, Y., and VAZIRGIANNIS, M., "Cluster validity methods: Part I and II," *SIGMOD Record*, vol. 31, no. 2, pp. 40–45, 2002.

[57] HASTIE, T., TIBSHIRANI, R., and FRIEDMANN, J., *The Elements of Statistical Learning*. Springer-Verlag, 2001.

[58] HINNEBURG, A. and KEIM, D. A., "An efficient approach to clustering in large multimedia databases with noise," *Proc. of ACM SIGKDD Conference*, pp. 58–65, 1998.

[59] HINNEBURG, A., KEIM, D. A., and WAWRYNIUK, M., "Visual mining of high-dimensional data," *IEEE Computer Graphics and Applications*, pp. 1–8, 1999.

[60] HOFFMAN, P., GRINSTEIN, G., MARX, K., GROSSE, I., and STANLEY, E., "Dna visual and analytic data mining," *IEEE Visualization*, pp. 437–442, 1997.

[61] HUANG, Z., DU, W., and CHEN, B., "Deriving private information from randomized data," *Proc. of ACM SIGMOD Conference*, 2005.

[62] HUANG, Z., "A fast clustering algorithm to cluster very large categorical data sets in data mining," *Workshop on Research Issues on Data Mining and Knowledge Discovery*, 1997.

[63] HYVARINEN, A., KARHUNEN, J., and OJA, E., *Independent Component Analysis*. Wiley-Interscience, 2001.

[64] INSELBERG, A., "Multidimensional detective," *IEEE Symposium on Information Visualization*, pp. 100–107, 1997.

[65] JAIN, A. K. and DUBES, R. C., *Algorithms for Clustering Data*. New York, USA: Prentice hall, 1988.

[66] JAIN, A. K. and DUBES, R. C., "Data clustering: A review," *ACM Computing Surveys*, vol. 31, pp. 264–323, 1999.

[67] JIANG, T., "How many entries in a typical orthogonal matrix can be approximated by independent normals," *To appear in The Annals of Probability*, 2005.

[68] KANDOGAN, E., "Visualizing multi-dimensional clusters, trends, and outliers using star coordinates," *Proc. of ACM SIGKDD Conference*, pp. 107–116, 2001.

[69] KARGUPTA, H., DATTA, S., WANG, Q., and SIVAKUMAR, K., "On the privacy preserving properties of random data perturbation techniques," *Proc. of Intl. Conf. on Data Mining (ICDM)*, 2003.

[70] KARYPIS, G., HAN, E.-H. S., and KUMAR, V., "Chameleon: hierarchical clustering using dynamic modeling," *IEEE Computer*, vol. 32, no. 8, pp. 68–75, 1999.

[71] KAUFMAN, C., PERLMAN, R., and SPECINER, M., *Network Security*. Prentice Hall, 2002.

[72] KEIM, D., "Visual exploration of large data sets," *ACM Communication*, vol. 44, no. 8, pp. 38–44, 2001.

[73] LARKIN, J. and SIMON, H., "Why a diagram is (sometimes) worth ten thousand words," *Cognitive Science*, vol. 11, pp. 65–99, 1987.

[74] LEBLANC, J., WARD, M. O., and WITTELS, N., "Exploring n-dimensional databases," *IEEE Conference on Visualization*, pp. 230–239, 1990.

[75] LEFEVRE, K., DEWITT, D. J., and RAMAKRISHNAN, R., "Mondrain multidimensional k-anonymity.," *Proc. of IEEE Intl. Conf. on Data Eng. (ICDE)*, 2006.

[76] LEHMANN, E. L. and CASELLA, G., *Theory of Point Estimation*. Springer-Verlag, 1998.

[77] LI, C., CHANG, E., GARCIA-MOLINA, H., and WIEDERHOLD, G., "Clustering for approximate similarity search in high-dimensional spaces," *IEEE Trans. on Knowledge and Data Eng.*, vol. 14, no. 4, pp. 792–808, 2002.

[78] LI, T., MA, S., and OGIHARA, M., "Entropy-based criterion in categorical clustering," *Proc. of Intl. Conf. on Machine Learning (ICML)*, 2004.

[79] LINDELL, Y. and PINKAS, B., "Privacy preserving data mining," *Journal of Cryptology*, vol. 15, no. 3, pp. 177–206, 2000.

[80] LITTLEFIELD, R. J., "Using the GLYPH concept to create user-definable display formats," *National Computer Graphics Association*, pp. 697–706, 1983.

[81] MACHANAVAJJHALA, A., GEHRKE, J., KIFER, D., and VENKITASUBRAMANIAM, M., "l-diversity: Privacy beyond k-anonymity," *Proc. of IEEE Intl. Conf. on Data Eng. (ICDE)*, 2006.

[82] MEEK, C., THIESSON, B., and HECKERMAN, D., "The learning-curve sampling method applied to model-based clustering.," *Journal of Machine Learning Research*, vol. 2, pp. 397–418, 2002.

[83] NETER, J., KUTNER, M. H., NACHTSHEIM, C. J., and WASSERMAN, W., *Applied Linear Statistical Methods*. WCB/McGraw-Hill, 1996.

[84] NEWMAN, M. E., "The structure and function of complex networks," *SIAM Review*, pp. 167–256, 2003.

[85] PALMER, C. and FALOUTSOS, C., "Density biased sampling: An improved method for data mining and clustering," *Proc. of ACM SIGMOD Conference*, pp. 82–92, 2000.

[86] QUINLAN, R., *C4.5: Programs form Machine Learning*. Morgan Kaufmann, 1993.

[87] RAMASWAMY, L., GEDIK, B., and LIU, L., "A distributed approach to node clustering in decentralized peer-to-peer networks," *IEEE Transactions on Parallel and Distributed Systems (TPDS)*, vol. 16, no. 9, pp. 1–16, Sept. 2005.

[88] ROSS, S. M., *Introduction to Probability Models*. San Diego, USA: Academic Press, 2000.

[89] ROUSSINOV, D., TOLLE, K., RAMSEY, M., and CHEN, H., "Interactive internet search through automatic clustering: an empirical study," *Proc. of ACM SIGIR Conference*, pp. 289–290, 1999.

[90] SADUN, L., *Applied Linear Algebra: the Decoupling Principle*. Prentice Hall, 2001.

[91] SCHUTZE, H. and SILVERSTEIN, C., "Projections for efficient document clustering," *Proc. of ACM SIGIR Conference*, pp. 74–81, 1997.

[92] SEO, J. and SHNEIDERMAN, B., "Interactively exploring hierarchical clustering results," *IEEE Computer*, vol. 35, no. 7, pp. 80–86, 2002.

[93] SHARMA, S., *Applied Multivariate Techniques.* New York, USA: Wiley&Sons, 1995.

[94] SHEIKHOLESLAMI, G., CHATTERJEE, S., and ZHANG, A., "Wavecluster: A multi-resolution clustering approach for very large spatial databases," *Proc. of Very Large Databases Conference (VLDB)*, pp. 428–439, 1998.

[95] SHNEIDERMAN, B., "Inventing discovery tools: Combining information visualization with data mining," *Information Visualization*, vol. 1, pp. 5–12, 2002.

[96] SILVERSTEIN, C. and PEDERSEN, J. O., "Almost-constant-time clustering of arbitrary corpus subsets," *Proc. of ACM SIGIR Conference*, pp. 60–66, 1997.

[97] SONKA, M., HLAVAC, V., and BOYLE, R., *Image Processing, Analysis and Machine Vision.* Pacific Grove, CA: Brooks/Cole Publishing, 1999.

[98] SPENCER, R., *Information Visualization.* US: Addison Wesley, 2000.

[99] STEWART, G., "The efficient generation of random orthogonal matrices with an application to condition estimation," *SIAM Journal on Numerical Analysis*, vol. 17, 1980.

[100] SWEENEY, L., "k-anonymity: a model for protecting privacy," *International Journal on Uncertainty, Fuzziness and Knowledge-based Systems*, vol. 10, no. 5, 2002.

[101] TISHBY, N., PEREIRA, F. C., and BIALEK, W., "The information bottleneck method," *Proc. of the 37-th Annual Allerton Conference on Communication, Control and Computing*, 1999.

[102] VAIDYA, J. and CLIFTON, C., "Privacy preserving association rule mining in vertically partitioned data," *Proc. of ACM SIGKDD Conference*, 2002.

[103] VAIDYA, J. and CLIFTON, C., "Privacy preserving k-means clustering over vertically partitioned data," *Proc. of ACM SIGKDD Conference*, 2003.

[104] VITTER, J. S., "Random sampling with a reservoir," *ACM Trans. on Mathematical Software*, vol. 11, no. 1, pp. 37–57, 1985.

[105] WARD, M., "Xmdvtool: Integrating multiple methods for visualizing multivariate data," *IEEE Visualization*, pp. 326–333, 1994.

[106] WILLETT, P., "Recent trends in hierarchic document clustering: A critical review," *Information Processing and Management*, vol. 24, no. 5, pp. 577–597, 1988.

[107] WRIGLEY, N., *Categorical Data Analysis for Geographers and Environmental Scientists*. Longman, 1985.

[108] XU, X., ESTER, M., KRIEGEL, H.-P., and SANDER, J., "A distribution-based clustering algorithm for mining in large spatial databases," *Proc. of IEEE Intl. Conf. on Data Eng. (ICDE)*, pp. 324–331, 1998.

[109] YANG, J., WARD, M. O., and RUNDENSTEINER, E. A., "Interactive hierarchical displays: a general framework for visualization and exploration of large multivariate datasets," *Computers and Graphics Journal*, vol. 27, pp. 265–283, 2002.

[110] YANG, L., "Interactive exploration of very large relational datasets through 3d dynamic projections," *Proc. of ACM SIGKDD Conference*, pp. 236–243, 2000.

[111] YANG, Y. and PEDERSEN, J. O., "A comparative study on feature selection in text categorization," *Proceedings of ICML-97, 14th International Conference on Machine Learning*, pp. 412–420, 1997.

[112] ZAMIR, O. and ETZIONI, O., "Web document clustering: a feasibility demonstration," *Proc. of ACM SIGIR Conference*, pp. 46–54, 1998.

[113] ZHANG, N., WANG, S., and ZHAO, W., "A new scheme on privacy-preserving data classification," *Proc. of ACM SIGKDD Conference*, 2005.

[114] ZHANG, T., RAMAKRISHNAN, R., and LIVNY., M., "BIRCH: An efficient data clustering method for very large databases," *Proc. of ACM SIGMOD Conference*, pp. 103–114, 1996.

VITA

Figure 112: Keke and his wife Yun.

Keke Chen was born in Wenzhou, Zhejiang, China. From 2001 to 2006, Keke Chen was a Ph.D. student in the College of Computing at Georgia Tech. Keke's research focuses on databases, data mining, data privacy protection, biomedical literature mining, and information visualization. Keke has been a research assistant with the Distributed Data Intensive Lab under the supervision of Dr. Ling Liu since 2002. He has published over ten papers in journals and conferences, and developed several research prototype systems. Before joining Georgia Tech, he was a lecturer in the department of Computer Science and Engineering at Zhejiang University from 1999 to 2000. Keke Chen received his M.S. degree in Computer Science in 1999 from Zhejiang University, China, and B.S. degree in Computer Science in 1996 from Tongji University, China.

Keke likes reading, fishing, and traveling in his spare time. In 2003, he married Ms. Yun Xing, who obtained Ph.D. degree in Biomedical Engineering from Georgia Tech in 2005.